Systèmes de recommandation avec Python

Bienvenue dans le notebook de code pour les systèmes de recommandation avec Python. Dans cette conférence, nous développerons des systèmes de recommandation de base à l'aide de Python et de pandas.

Dans ce projet, nous nous concentrerons sur la fourniture d'un système de recommandation de base en suggérant des éléments qui ressemblent le plus à un élément particulier, dans ce cas, les films. Gardez à l'esprit qu'il ne s'agit pas d'un véritable système de recommandation robuste, pour le décrire plus précisément, il vous indique simplement quels films/éléments sont les plus similaires à votre choix de film.

Importation des bibliothèques

In [2]:
```
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

Importation des données

In [3]:
```
column_names = ['user_id', 'item_id', 'rating', 'timestamp']
df = pd.read_csv('u.data', sep='\t', names=column_names)
```

In [4]:
```
df.head()
```

Out[4]:

	user_id	item_id	rating	timestamp
0	0	50	5	881250949
1	0	172	5	881250949
2	0	133	1	881250949
3	196	242	3	881250949
4	186	302	3	891717742

Prenons maintenant les titres des films :

In [5]:
```
movie_titles = pd.read_csv("Movie_Id_Titles")
movie_titles.head()
```

Out[5]:

	item_id	title
0	1	Toy Story (1995)
1	2	GoldenEye (1995)
2	3	Four Rooms (1995)
3	4	Get Shorty (1995)
4	5	Copycat (1995)

Nous pouvons les fusionner ensemble :

In [6]:
```
df = pd.merge(df,movie_titles,on='item_id')
df.head()
```

Out[6]:

	user_id	item_id	rating	timestamp	title
0	0	50	5	881250949	Star Wars (1977)
1	0	172	5	881250949	Empire Strikes Back, The (1980)
2	0	133	1	881250949	Gone with the Wind (1939)
3	196	242	3	881250949	Kolya (1996)
4	186	302	3	891717742	L.A. Confidential (1997)

EDA

Explorons les données en jetant un coup d'œil à certains des films les mieux notés.

Visualisation

In [7]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('white')
%matplotlib inline
```

Créons un dataframe d'évaluations avec une note moyenne et le nombre d'évaluations :

In [8]:
```
df.groupby('title')['rating'].mean().sort_values(ascending=False).head()
```

Out[8]:
```
title
They Made Me a Criminal (1939)                5.0
Marlene Dietrich: Shadow and Light (1996)     5.0
Saint of Fort Washington, The (1993)          5.0
Someone Else's America (1995)                 5.0
Star Kid (1997)                               5.0
Name: rating, dtype: float64
```

In [9]:
```
df.groupby('title')['rating'].count().sort_values(ascending=False).head()
```

Out[9]:
```
title
Star Wars (1977)             584
Contact (1997)               509
Fargo (1996)                 508
Return of the Jedi (1983)    507
Liar Liar (1997)             485
Name: rating, dtype: int64
```

In [10]:
```
ratings = pd.DataFrame(df.groupby('title')['rating'].mean())
ratings.head()
```

Out[10]:

title	rating
'Til There Was You (1997)	2.333333
1-900 (1994)	2.600000
101 Dalmatians (1996)	2.908257
12 Angry Men (1957)	4.344000
187 (1997)	3.024390

Définissons maintenant la colonne du nombre d'évaluations :

In [11]:
```
ratings['num of ratings'] = pd.DataFrame(df.groupby('title')['rating'].count
ratings.head()
```

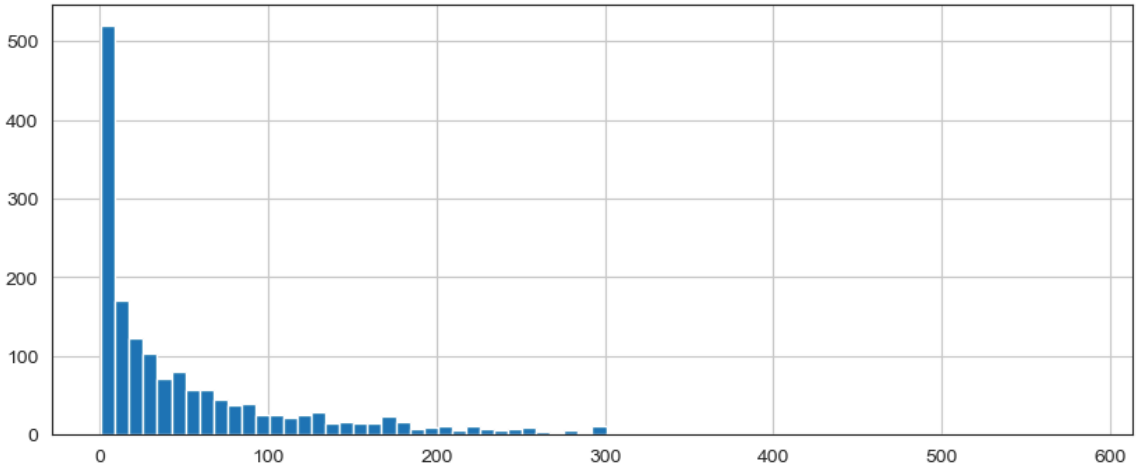
Out[11]:

title	rating	num of ratings
'Til There Was You (1997)	2.333333	9
1-900 (1994)	2.600000	5
101 Dalmatians (1996)	2.908257	109
12 Angry Men (1957)	4.344000	125
187 (1997)	3.024390	41

Maintenant, quelques histogrammes :

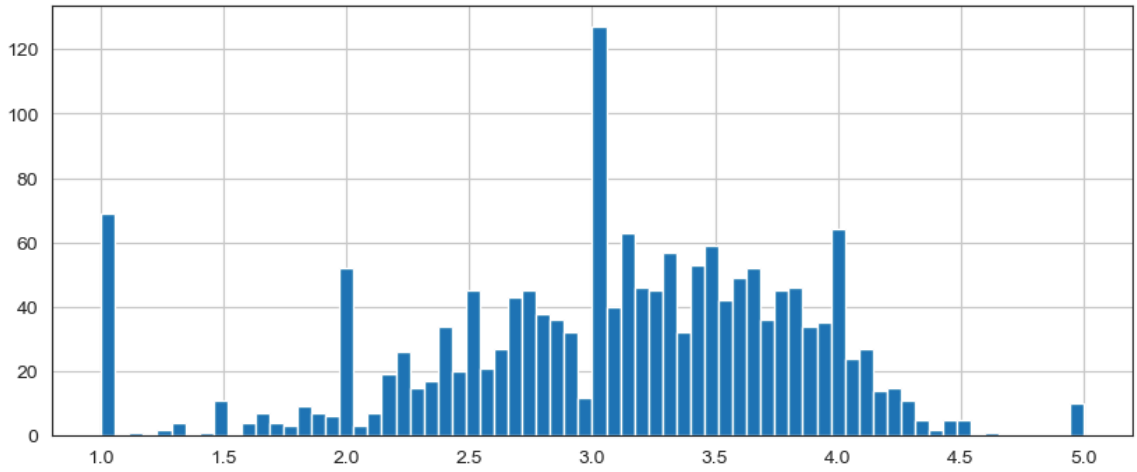
In [12]:
```
plt.figure(figsize=(10,4))
ratings['num of ratings'].hist(bins=70)
```

Out[12]: <Axes: >



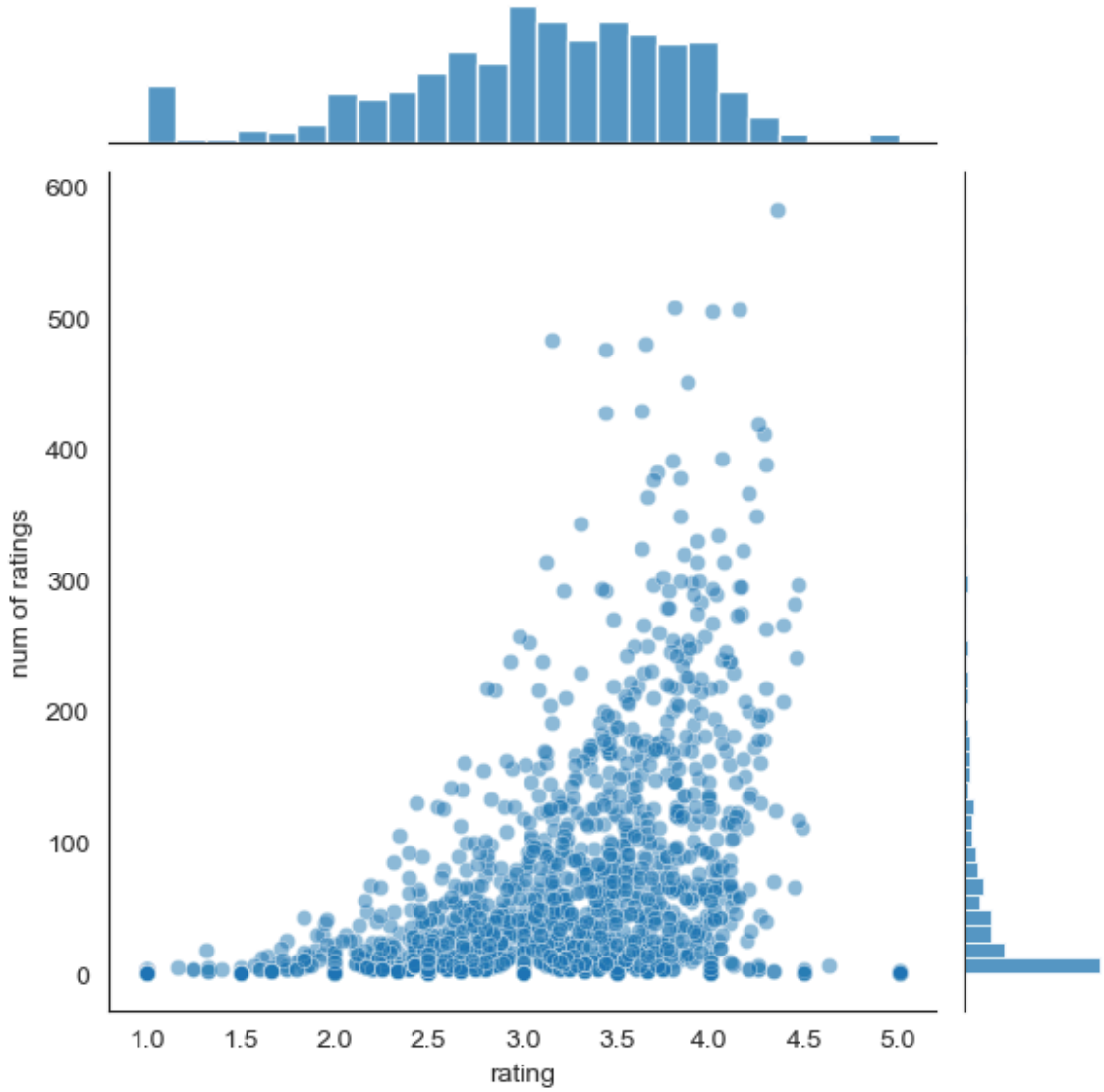
In [13]:
```
plt.figure(figsize=(10,4))
ratings['rating'].hist(bins=70)
```

Out[13]: <Axes: >



In [14]:
```
sns.jointplot(x='rating',y='num of ratings',data=ratings,alpha=0.5)
```

Out[14]:
```
<seaborn.axisgrid.JointGrid at 0x1f2aed39150>
```



D’accord! Maintenant que nous avons une idée générale de ce à quoi ressemblent les données, passons à la création d’un système de recommandation simple :

Recommander des films similaires

Créons maintenant une matrice qui a les ID utilisateur sur un accès et le titre du film sur un autre axe. Chaque cellule sera alors constituée de la note que l’utilisateur a donnée à ce film. Notez qu’il y aura beaucoup de valeurs NaN, car la plupart des gens n’ont pas vu la plupart des films.

In [15]:
```
moviemat = df.pivot_table(index='user_id',columns='title',values='rating')
moviemat.head()
```

Out[15]:

user_id	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: A Space Odyssey (1968)	3 Ninjas: High Noon At Mega Mountain (1998)	Step Tl (193
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
1	NaN	NaN	2.0	5.0	NaN	NaN	3.0	4.0	NaN	Na
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0	Na
3	NaN	NaN	NaN	NaN	2.0	NaN	NaN	NaN	NaN	Na
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

5 rows × 1664 columns

Most rated movie:

In [16]:
```
ratings.sort_values('num of ratings',ascending=False).head(10)
```

Out[16]:

title	rating	num of ratings
Star Wars (1977)	4.359589	584
Contact (1997)	3.803536	509
Fargo (1996)	4.155512	508
Return of the Jedi (1983)	4.007890	507
Liar Liar (1997)	3.156701	485
English Patient, The (1996)	3.656965	481
Scream (1996)	3.441423	478
Toy Story (1995)	3.878319	452
Air Force One (1997)	3.631090	431
Independence Day (ID4) (1996)	3.438228	429

Let's choose two movies: starwars, a sci-fi movie. And Liar Liar, a comedy.

In [17]:
```
ratings.head()
```

Out[17]:

title	rating	num of ratings
'Til There Was You (1997)	2.333333	9
1-900 (1994)	2.600000	5
101 Dalmatians (1996)	2.908257	109
12 Angry Men (1957)	4.344000	125
187 (1997)	3.024390	41

Voyons maintenant les notes des utilisatrices pour ces deux films :

In [18]:
```
starwars_user_ratings = moviemat['Star Wars (1977)']
liarliar_user_ratings = moviemat['Liar Liar (1997)']
starwars_user_ratings.head()
```

Out[18]:
```
user_id
0    5.0
1    5.0
2    5.0
3    NaN
4    5.0
Name: Star Wars (1977), dtype: float64
```

Nous pouvons ensuite utiliser la méthode corrwith() pour obtenir des corrélations entre deux séries de pandas :

In [19]:
```
similar_to_starwars = moviemat.corrwith(starwars_user_ratings)
similar_to_liarliar = moviemat.corrwith(liarliar_user_ratings)
```

Nettoyons cela en supprimant les valeurs NaN et en utilisant un DataFrame au lieu d'une série :

In [20]:
```
corr_starwars = pd.DataFrame(similar_to_starwars,columns=['Correlation'])
corr_starwars.dropna(inplace=True)
corr_starwars.head()
```

Out[20]:

title	Correlation
'Til There Was You (1997)	0.872872
1-900 (1994)	-0.645497
101 Dalmatians (1996)	0.211132
12 Angry Men (1957)	0.184289
187 (1997)	0.027398

Maintenant, si nous trions la trame de données par corrélation, nous devrions obtenir les films les plus similaires, mais notez que nous obtenons des résultats qui n'ont pas vraiment de sens. En effet, il y a beaucoup de films qui ne sont regardés qu'une seule fois par des

utilisateurs qui ont également regardé Star Wars (c'était le film le plus populaire).

In [21]:
```
corr_starwars.sort_values('Correlation',ascending=False).head(10)
```

Out[21]:

title	Correlation
Commandments (1997)	1.0
Cosi (1996)	1.0
No Escape (1994)	1.0
Stripes (1981)	1.0
Man of the Year (1995)	1.0
Hollow Reed (1996)	1.0
Beans of Egypt, Maine, The (1994)	1.0
Good Man in Africa, A (1994)	1.0
Old Lady Who Walked in the Sea, The (Vieille qui marchait dans la mer, La) (1991)	1.0
Outlaw, The (1943)	1.0

Let's fix this by filtering out movies that have less than 100 reviews (this value was chosen based off the histogram from earlier).

In [22]:
```
corr_starwars = corr_starwars.join(ratings['num of ratings'])
corr_starwars.head()
```

Out[22]:

title	Correlation	num of ratings
'Til There Was You (1997)	0.872872	9
1-900 (1994)	-0.645497	5
101 Dalmatians (1996)	0.211132	109
12 Angry Men (1957)	0.184289	125
187 (1997)	0.027398	41

Maintenant, triez les valeurs et remarquez comment les titres ont beaucoup plus de sens :

In [23]:
```
corr_starwars[corr_starwars['num of ratings']>100].sort_values('Correlation'
```

Out[23]:

title	Correlation	num of ratings
Star Wars (1977)	1.000000	584
Empire Strikes Back, The (1980)	0.748353	368
Return of the Jedi (1983)	0.672556	507
Raiders of the Lost Ark (1981)	0.536117	420
Austin Powers: International Man of Mystery (1997)	0.377433	130

Maintenant, c'est la même chose pour la comédie Liar Liar :

In [24]:
```
corr_liarliar = pd.DataFrame(similar_to_liarliar,columns=['Correlation'])
corr_liarliar.dropna(inplace=True)
corr_liarliar = corr_liarliar.join(ratings['num of ratings'])
corr_liarliar[corr_liarliar['num of ratings']>100].sort_values('Correlation'
```

Out[24]:

title	Correlation	num of ratings
Liar Liar (1997)	1.000000	485
Batman Forever (1995)	0.516968	114
Mask, The (1994)	0.484650	129
Down Periscope (1996)	0.472681	101
Con Air (1997)	0.469828	137

Type *Markdown* and LaTeX: α^2

In [ ]: