9.8 Surprise 패키지

민소연 차수빈

EWHA EURON

1. Surprise 패키지 소개

EWHA EURON

1. Surprise 패키지 소개

●Surprise 패키지

- 파이썬 기반 추천 시스템 구축을 위한 전용 패키지
  - 사이킷런은 추천 전용 모델 제공 X
- API를 이용해 쉽게 추천 시스템을 구축할 수 있게 만들어짐

●장점

- 다양한 추천 알고리즘(사용자 또는 아이템 기반 최근접 이웃 협업 필터링, SVD, SVD++, NMF) 기반의 잠재 요인 협업 필터링을 쉽게 적용해 추천 시스템을 구축할 수 있음
- Surprise의 핵심 API는 사이킷런의 핵심 API와 유사한 API명으로 작성됨
  - fit( ), predict( ), train_test_split( ), cross_validate( ), GridSearchCV 클래스 등

2. Surprise를 이용한 추천 시스템 구축

EWHA EURON

2. Surprise를 이용한 추천 시스템 구축

User ID	Item ID	Rating
User 1	Item1	3
User 1	Item3	3
User 2	Item1	4

●예제 실습

- Surprise에서 데이터 로딩은 Dataset 클래스를 이용해서만 가능하다.
- 주요 데이터가 로우(Row) 레벨 형태로 되어 있는 포맷의 데이터만 처리한다.

```
import surprise
from surprise import SVD
from surprise import Dataset
from surprise import accuracy
from surprise.model_selection import train_test_split
```

load_builtin( ) : 무비렌즈 사이트에서 제공하는 과거 버전 데이터셋인 'ml-100k' (10만 개 평점 데이터)를 아카이브 세트로부터 내려받아 로컬 디렉터리에 저장한 뒤 데이터를 로딩함.

```
data = Dataset.load_builtin('ml-100k')
# 수행 시마다 동일하게 데이터를 분할하기 위해 random_state 값 부여
trainset, testset = train_test_split(data, test_size=.25, random_state=0)
```

```
Dataset ml-100k could not be found. Do you want to download it? [Y/n]
Y
```

처음에는 로컬 디렉터리에 데이터가 없어서 무비렌즈 사이트에서 내려받을 것인지 물어봄. 이후에는 무비렌즈 접속 없이 저장된 데이터셋 로딩

2. Surprise를 이용한 추천 시스템 구축

●예제 실습

- 추천을 예측하는 메서드: test( ), predict( )
- **test( )** : 사용자-아이템 평점 데이터세트 전체에 대해서 추천을 예측하는 메서드
  즉, 입력된 데이터 세트에 대해서 추천 데이터세트를 만들어줌.
- **predict( )** : 개별 사용자와 영화에 대한 추천 평점을 반환해줌.

```
algo = SVD(random_state=0) # 알고리즘 객체 생성
algo.fit(trainset)
```

- test( ) 메서드 호출 결과
  - 파이썬 리스트
  - 크기는 입력 인자 데이터셋과 같은 25,000개 Prediction 객체

```
predictions = algo.test( testset )
print('prediction type :',type(predictions), ' size:',len(predictions))
print('prediction 결과의 최초 5개 추출')
predictions[:5]
```

```
prediction type : <class 'list'>  size: 25000
prediction 결과의 최초 5개 추출
[Prediction(uid='120', iid='282', r_ui=4.0, est=3.5114147666251547, details=
{'was_impossible': False}),
 Prediction(uid='882', iid='291', r_ui=4.0, est=3.573872419581491, details=
{'was_impossible': False}),
 Prediction(uid='535', iid='507', r_ui=5.0, est=4.033583485472447, details=
{'was_impossible': False}),
 Prediction(uid='697', iid='244', r_ui=5.0, est=3.8463639495936905, details=
{'was_impossible': False}),
 Prediction(uid='751', iid='385', r_ui=4.0, est=3.1807542478219157, details=
{'was_impossible': False})]
```

2. Surprise를 이용한 추천 시스템 구축

●예제 실습

- Prediction 객체
  - Surprise 패키지에서 제공하는 데이터 타입
  - 개별 사용자 아이디(uid), 영화(또는 아이템) 아이디(iid)와 실제 평점(r_ui) 정보에 기반해 Surprise의 추천 예측 평점(est) 데이터를 튜플 형태로 가짐.

```
algo = SVD(random_state=0) # 알고리즘 객체 생성
algo.fit(trainset)
```

```
predictions = algo.test( testset )
print('prediction type :',type(predictions), ' size:',len(predictions))
print('prediction 결과의 최초 5개 추출')
predictions[:5]
```

```
prediction type : <class 'list'>  size: 25000
prediction 결과의 최초 5개 추출
[Prediction(uid='120', iid='282', r_ui=4.0, est=3.5114147666251547, details=
{'was_impossible': False}),
 Prediction(uid='882', iid='291', r_ui=4.0, est=3.573872419581491, details=
{'was_impossible': False}),
 Prediction(uid='535', iid='507', r_ui=5.0, est=4.033583485472447, details=
{'was_impossible': False}),
 Prediction(uid='697', iid='244', r_ui=5.0, est=3.8463639495936905, details=
{'was_impossible': False}),
 Prediction(uid='751', iid='385', r_ui=4.0, est=3.1807542478219157, details=
{'was_impossible': False})]
```

  - Predictions 객체의 details 속성은 내부 처리 시 추천 예측을 할 수 없는 경우에 로그용으로 데이터를 남기는 데 사용
  - was_impossible이 True : 예측값을 생성할 수 없는 데이터

2. Surprise를 이용한 추천 시스템 구축

●예제 실습

- Prediction 객체
  - uid, iid, r_ui, est 등의 속성에 접근하기 위해선 객체명.uid와 같은 형식 사용

```
[ (pred.uid, pred.iid, pred.est) for pred in predictions[:3] ]
```

```
[('120', '282', 3.5114147666251547),
 ('882', '291', 3.5738724195814 91),
 ('535', '507', 4.033583485472447)]
```

- Predict( ) 메서드 호출 결과

```
# 사용자 아이디, 아이템 아이디는 문자열로 입력해야 함.
uid = str(196)
iid = str(302)
pred = algo.predict(uid, iid)
print(pred)
```

```
user: 196        item: 302        r_ui = None   est = 4.49   {'was_impossible': False}
```

- 개별 사용자의 아이템에 대한 추천 평점 예측
- 인자로 개별 사용자 아이디, 아이템 아이디를 문자열로 입력하면 추천 예측 평점을 포함한 정보를 반환
- test( ) 메서드는 모든 사용자와 아이템 아이디에 대해 predict( ) 를 반복적으로 수행한 결과라고 생각하면 된다.

2. Surprise를 이용한 추천 시스템 구축

●예제 실습

- accuracy 모듈
  - RMSE, MSE 등의 방법으로 추천 시스템의 성능 평가 정보를 제공한다.

```
accuracy.rmse(predictions)
```

```
RMSE: 0.9467
0.9466860806937948
```

```
accuracy.mse(predictions)
```

```
MSE: 0.8962
0.8962145353793782
```

3. Surprise 주요 모듈 소개

EWHA EURON

3. Surprise 주요 모듈 소개

●Dataset

- Surprise는 user_id(사용자 아이디), item_id(아이템 아이디), rating(평점) 데이터가 로우 레벨로 된 데이터세트만 적용할 수 있다.
- 첫 번째 칼럼을 사용자 아이디, 두 번째 칼럼을 아이템 아이디, 세 번째 칼럼을 평점으로 가정해 데이터를 로딩하고 네 번째 칼럼부터는 아예 로딩을 수행하지 않는다.
- 무비렌즈 아카이브 서버에서 자동으로 내려받는 데이터 파일뿐만 아니라 일반 데이터 파일이나 판다스 DataFrame에서도 로딩할 수 있다.
- 단, 데이터 세트의 칼럼 순서가 반드시 사용자 아이디, 아이템 아이디, 평점 순으로 되어있어야 한다.

3. Surprise 주요 모듈 소개

●Dataset

API 명	내용
Dataset.load_builtin (name=' ml-100k')	- 무비렌즈 아카이브 FTP 서버에서 무비렌즈 데이터를 내려받음 - ml-100k, ml-1M를 내려받을 수 있습니다. 일단 내려받은 데이터는 .surprise_data 디렉터리 밑에 저장되고, 해당 디렉터리에 데이터가 있으면 FTP에서 내려받지 않고 해당 데이터를 이용 - 입력 파라미터인 name으로 대상 데이터가 ml-100k인지 ml-1m인지를 입력 (name='ml-100k' , default = ml-100k)
Dataset.load_from_file (file_path, reader)	- OS 파일에서 데이터를 로딩할 때 사용 - 콤마, 탭 등으로 칼럼이 분리된 포맷의 OS 파일에서 데이터 load - 입력 파라미터로 OS 파일명, Reader로 파일의 포맷을 지정

3. Surprise 주요 모듈 소개

●Dataset

API 명	내용
Dataset.load_from_df (df, reader)	- 판다스의 DataFrame을 데이터로 load - 파라미터로 DataFrame을 입력받으며 DataFrame 역시 반드시 3개의 칼럼인 사용자 아이디, 아이템 아이디, 평점 순으로 칼럼 순서가 정해져 있어야 함 - 입력 파라미터로 DataFrame 객체, Reader로 파일의 포맷을 지정합니다.

3. Surprise 주요 모듈 소개

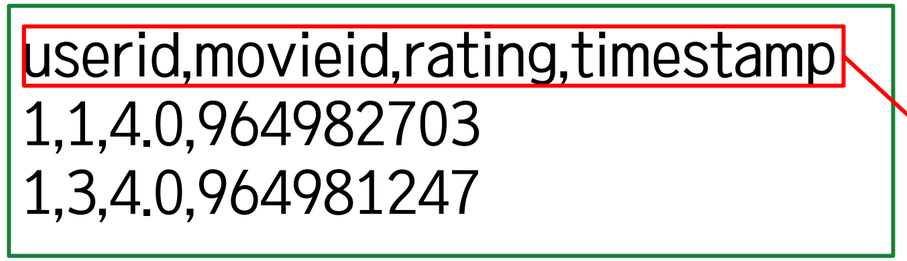
●OS 파일 데이터를 Surprise 데이터 세트로 로딩

○.load_from_file API

주의할 점

- 로딩되는 데이터 파일에 칼럼명을 가지는 헤더 문자열이 있어서는 X

  => pd.DataFrame.to_csv( ) 함수를 이용해 칼럼 헤더 삭제



rating.csv - 칼럼 Header 제거 필요

```
import pandas as pd

ratings = pd.read_csv('/content/drive/MyDrive/EURON/ml-latest-small/ratings.csv')
# ratings_noh.csv 파일로 unload 시 index 와 header를 모두 제거한 새로운 파일 생성.
ratings.to_csv('/content/drive/MyDrive/EURON/ml-latest-small/ratings_noh.csv', index=False, header=False)
```

3. Surprise 주요 모듈 소개

●OS 파일 데이터를 Surprise 데이터 세트로 로딩

○.load_from_file API

Reader 클래스

- ratings_noh.csv는 칼럼 헤더가 없고, 4개의 칼럼이 콤마로만 분리되어 있음

  -> 데이터 로딩 시 Rating 클래스로 파싱 포맷 정의

- Surprise 데이터 세트는 기본적으로 무비렌즈 데이터 형식을 따름

  -> 무비렌즈 데이터 형식이 아닌 다른 OS 파일인 경우 Reader 클래스를 먼저 설정해야 함

```
from surprise import Reader

reader = Reader(line_format='user item rating timestamp', sep=',', rating_scale=(0.5, 5))
data=Dataset.load_from_file('/content/drive/MyDrive/EURON/ml-latest-small/ratings_noh.csv',reader=reader)
```

3. Surprise 주요 모듈 소개

●OS 파일 데이터를 Surprise 데이터 세트로 로딩

○.load_from_file API

Reader 클래스

- 주요 생성 파라미터

파라미터 명	내용
line_format (string)	- 칼럼을 순서대로 나열 - 입력된 문자열을 공백으로 분리해 칼럼으로 인식
sep (char)	- 칼럼을 분리하는 분리자(default = '\t') - 판다스 DataFrame에서 입력받을 경우에는 기재할 필요 x
rating_scale (tuple, optional)	- 평점 값의 최소 ~ 최대 평점을 설정(default = (1, 5)) - ratings.csv 파일의 경우는 최소 평점이 0.5, 최대 평점이 5이므로 (0.5, 5)로 설정

3. Surprise 주요 모듈 소개

●OS 파일 데이터를 Surprise 데이터 세트로 로딩

○.load_from_file API

SVD 행렬 분해 기법을 이용해 추천 예측

```
trainset, testset = train_test_split(data, test_size=.25, random_state=0)

# 수행시마다 동일한 결과 도출을 위해 random_state 설정
algo = SVD(n_factors=50, random_state=0)

# 학습 데이터 세트로 학습 후 테스트 데이터 세트로 평점 예측 후 RMSE 평가
algo.fit(trainset)
predictions = algo.test( testset )
accuracy.rmse(predictions)
```

```
RMSE: 0.8682
0.8681952927143516
```

3. Surprise 주요 모듈 소개

●판다스 DataFrame에서 Surprise 데이터 세트로 로딩

○.load_from_df API

주의할 점

- 칼럼 순서를 지켜야 함(사용자 아이디, 아이템 아이디, 평점)
- 다음과 같이 파라미터를 지정해주면 됨

```
import pandas as pd
from surprise import Reader, Dataset

reader = Reader(rating_scale=(0.5, 5.0))

# ratings DataFrame 에서 컬럼은 사용자 아이디, 아이템 아이디, 평점 순서를 지켜야 합니다.
data = Dataset.load_from_df(ratings[['userId', 'movieId', 'rating']], reader)
trainset, testset = train_test_split(data, test_size=.25, random_state=0)

algo = SVD(n_factors=50, random_state=0)
algo.fit(trainset)
predictions = algo.test( testset )
accuracy.rmse(predictions)
```

```
RMSE: 0.8682
0.8681952927143516
```

4. Surprise 추천 알고리즘 클래스

EWHA EURON

4. Surprise 추천 알고리즘 클래스

- 자주 사용되는 추천 알고리즘 클래스

클래스명	설명
SVD	행렬 분해를 통한 잠재 요인 협업 필터링을 위한 SVD 알고리즘
KNNBasic	최근접 이웃 협업 필터링을 위한 KNN 알고리즘
BaselineOnly	사용자 Bias와 아이템 Bias를 감안한 SGD 베이스라인 알고리즘

4. Surprise 추천 알고리즘 클래스

● SVD 클래스

- 비용 함수

  - 사용자 예측 Rating: $r\hat{}ui=\mu+bu+bi+qTipu$
  - Regularization을 적용한 비용 함수: $\Sigma(rui-r\hat{}ui)2+\lambda(b2i+b2u+||qi||2+||pu||2)$

- 입력 파라미터

파라미터명	설명
n_factors	- 잠재 요인 K의 개수 (default = 100) - 커질수록 정확도가 높아지나 overfitting 발생 위험 존재
n_epochs	- SGD(Stochastic Gradient Descent) 수행 시의 반복 횟수 (default = 20)
biased(bool)	- 베이스라인 사용자 편향 적용 여부 (deualt = True)

4. Surprise 추천 알고리즘 클래스

● 추천 알고리즘의 예측 성능 Benchmark

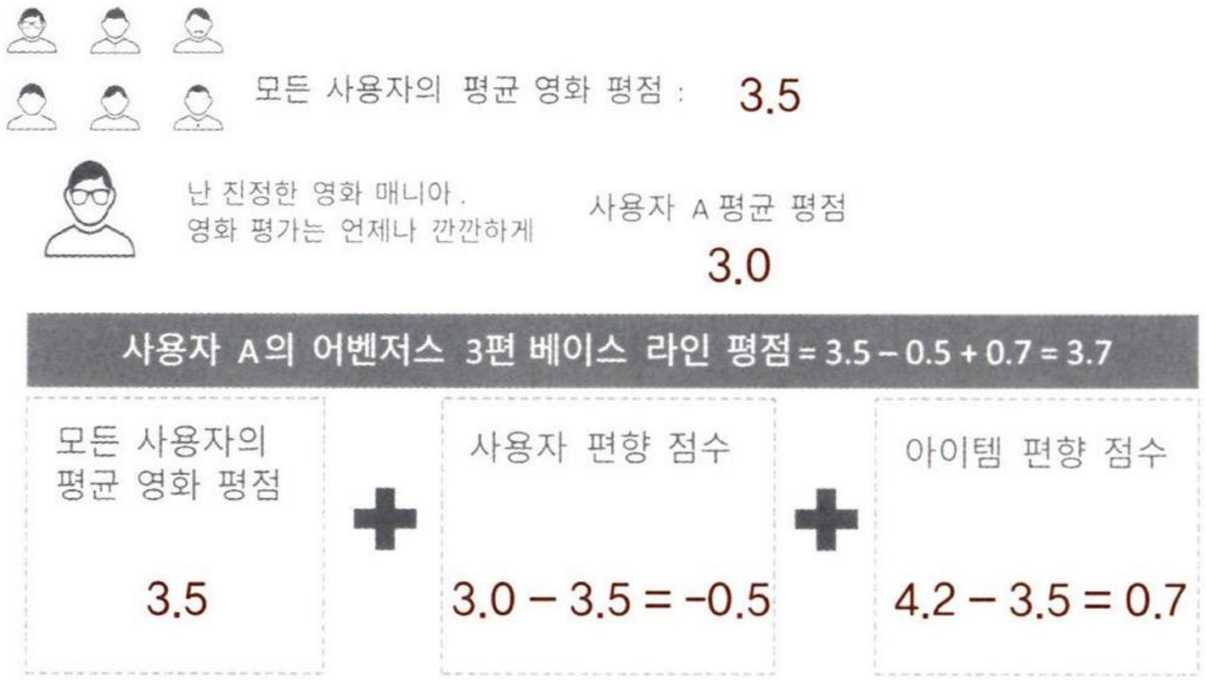
- http://surpriselib.com/

Movielens 100k	RMSE	MAE	Time
SVD	0.934	0.737	0:00:06
SVD++ (cache_ratings=False)	0.919	0.721	0:01:39
SVD++ (cache_ratings=True)	0.919	0.721	0:01:22
NMF	0.963	0.758	0:00:06
Slope One	0.946	0.743	0:00:09
k-NN	0.98	0.774	0:00:08
Centered k-NN	0.951	0.749	0:00:09
k-NN Baseline	0.931	0.733	0:00:13
Co-Clustering	0.963	0.753	0:00:06
Baseline	0.944	0.748	0:00:02
Random	1.518	1.219	0:00:01

Movielens 1M	RMSE	MAE	Time
SVD	0.873	0.686	0:01:07
SVD++ (cache_ratings=False)	0.862	0.672	0:41:06
SVD++ (cache_ratings=True)	0.862	0.672	0:34:55
NMF	0.916	0.723	0:01:39
Slope One	0.907	0.715	0:02:31
k-NN	0.923	0.727	0:05:27
Centered k-NN	0.929	0.738	0:05:43
k-NN Baseline	0.895	0.706	0:05:55
Co-Clustering	0.915	0.717	0:00:31
Baseline	0.909	0.719	0:00:19
Random	1.504	1.206	0:00:19

+ Baseline 평점

- 개인의 성향을 반영해 아이템 평가에 편향성 요소를 반영하여 평점을 부과하는 것
- 보통 전체 평균 평점 + 사용자 편향 점수 + 아이템 편향 점수로 계산됨
  - 전체 평균 평점 = 모든 사용자의 아이템에 대한 평점을 평균한 값
  - 사용자 편향 점수 = 사용자 별 아이템 평점 평균 값 - 전체 평균 평점
  - 아이템 편향 점수 = 아이템 별 평점 평균 값 - 전체 평균 평점



어벤저스 3편 평균 평점

4.2

5. 교차 검증과 하이퍼 파라미터 튜닝

EWHA EURON

5. 교차 검증과 하이퍼 파라미터 튜닝

- Surprise는 교차 검증과 하이퍼 파라미터 튜닝을 위해 사이킷런과 유사한 cross_validate() 와 GridSearchCV 클래스를 제공

- cross_validate() 함수
  - surprise.model_selection 모듈 내에 존재
  - fold 된 데이터 세트의 개수와 성능 측정 방법을 명시해 교차 검증을 수행
  - fold 별 성능 평가 수치와 전체 폴드의 평균 성능 평가 수치를 함께 보여줌

- GridSearchCV
  - 교차 검증을 통한 하이퍼 파라미터 최적화를 수행

5. 교차 검증과 하이퍼 파라미터 튜닝

```
from surprise.model_selection import cross_validate

# Pandas DataFrame에서 Surprise Dataset으로 데이터 로딩
ratings = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/                    /data/ratings.csv')
reader = Reader(rating_scale = (0.5, 5.0))
data = Dataset.load_from_df(ratings[['userId', 'movieId', 'rating']], reader)

algo = SVD(random_state = 0)
cross_validate(algo, data, measures = ['RMSE', 'MAE'], cv = 5, verbose = True)
```

```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.8858  0.8756  0.8644  0.8741  0.8784  0.8757  0.0069
MAE (testset)     0.6818  0.6735  0.6646  0.6700  0.6754  0.6731  0.0057
Fit time          1.43    1.43    1.44    1.41    1.72    1.48    0.12
Test time         0.13    0.11    0.12    0.11    0.21    0.14    0.04
{'test_rmse': array([0.88582553, 0.87561245, 0.8644275 , 0.87414696, 0.87837489]),
 'test_mae': array([0.68180608, 0.67353541, 0.66463134, 0.66996845, 0.67537429]),
 'fit_time': (1.4258153438568115,
  1.4305624961853027,
  1.4352211952209473,
  1.4143743515014648,
  1.7185025215148926),
 'test_time': (0.1271040439605713,
  0.11114096641540527,
  0.11785507202148438,
  0.11095142364501953,
  0.2091503143310547)}
```

5. 교차 검증과 하이퍼 파라미터 튜닝

```
from surprise.model_selection import GridSearchCV

# 최적화 할 파라미터들을 딕셔너리 형태로 지정
param_grid = {'n_epochs': [20, 40, 60], 'n_factors': [50, 100, 200]}

# CV를 3개 폴드 세트로 지정
# 성능 평가는 rmse, mse로 수행하도록 GridSearchCV 구성
gs = GridSearchCV(SVD, param_grid,
                  measures = ['rmse', 'mae'], cv = 3)
gs.fit(data)

# 최고 RMSE Evaluation 점수와 그때의 하이퍼 파라미터
print(gs.best_score['rmse'])
print(gs.best_params['rmse'])
```

```
0.8772018168830553
{'n_epochs': 20, 'n_factors': 50}
```

6. 개인화 영화 추천 시스템 구축

EWHA EURON

6. 개인화 영화 추천 시스템 구축

- **목표>** 학습된 추천 알고리즘을 기반으로 특정 사용자가 아직 평점을 매기지 않은 (= 관람하지 않은) 영화 중에서 개인 취향에 가장 적절한 영화를 추천

- Surprise는 데이터 세트를 train_test_split()을 이용해 내부에서 사용하는 TrainSet 클래스 객체로 변환하지 않으면 fit()을 통해 학습 불가능!

```
# train_test_split( )으로 분리되지 않는 Dataset에 fit()을 호출하면 오류 발생

data = Dataset.load_from_df(ratings[['userId', 'movieId', 'rating']], reader)
algo = SVD(n_factors=50, random_state=0)
algo.fit(data)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-21-33c08dace4bd> in <cell line: 4>()
      2 data = Dataset.load_from_df(ratings[['userId', 'movieId', 'rating']], reader)
      3 algo = SVD(n_factors=50, random_state=0)
----> 4 algo.fit(data)

 1 frames
/usr/local/lib/python3.10/dist-packages/surprise/prediction_algorithms/matrix_factorization.pyx in surprise.prediction_algorithms.matrix_factorization.SVD.sgd()

AttributeError: 'DatasetAutoFolds' object has no attribute 'n_users'
```

6. 개인화 영화 추천 시스템 구축

- 데이터 세트 **전체**를 학습 데이터로 사용하려면 DatasetAutoFolds 클래스를 활용
  - build_full_trainset() 메서드를 호출하면 전체 데이터를 학습 데이터 세트로 만들 수 있음

```
from surprise.dataset import DatasetAutoFolds

reader = Reader(line_format = 'user item rating timestamp', sep = ',', rating_scale = (0.5, 5))

# DatasetAutoFolds 클래스를 ratings_noh.csv 파일 기반으로 생성
data_folds = DatasetAutoFolds(ratings_file = './data/ratings_noh.csv', reader = reader)

# 전체 데이터를 학습데이터로 생성
trainset = data_folds.build_full_trainset()
```

6. 개인화 영화 추천 시스템 구축

```
### 추천 알고리즘 학습

algo = SVD(n_epochs = 20, n_factors = 50, random_state = 0)
algo.fit(trainset)
```

```
# 영화에 대한 상세 속성 정보 DataFrame 로딩
movies = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/                /data/movies.csv')

# userId = 9 의 movieId 데이터를 추출하여 movieId = 42 데이터가 있는지 확인
movieIds = ratings[ratings['userId'] == 9]['movieId']
if movieIds[movieIds==42].count() == 0:
    print('사용자 아이디 9는 영화 아이디 42의 평점 없음')

print(movies[movies['movieId']==42])
```

```
사용자 아이디 9는 영화 아이디 42의 평점 없음
    movieId                    title                genres
38       42  Dead Presidents (1995)  Action|Crime|Drama
```

```
uid = str(9)
iid = str(42)

pred = algo.predict(uid, iid, verbose=True)
```

```
user: 9          item: 42         r_ui = None   est = 3.13   {'was_impossible': False}
```

EWHA EURON

6. 개인화 영화 추천 시스템 구축

- 추천 대상이 되는 영화를 추출하는 함수

```
def get_unseen_surprise(ratings, movies, userId):
    # 입력값으로 들어온 userId에 해당하는 사용자가 평점을 매긴 모든 영화를 리스트로 생성
    seen_movies = ratings[ratings['userId']== userId]['movieId'].tolist()

    # 모든 영화들의 movieId를 리스트로 생성.
    total_movies = movies['movieId'].tolist()

    # 모든 영화들의 movieId중 이미 평점을 매긴 영화의 movieId를 제외하여 리스트로 생성
    unseen_movies= [movie for movie in total_movies if movie not in seen_movies]
    print('평점 매긴 영화수:',len(seen_movies), '추천대상 영화수:',len(unseen_movies), ₩
          '전체 영화수:',len(total_movies))

    return unseen_movies

unseen_movies = get_unseen_surprise(ratings, movies, 9)
```

```
평점 매긴 영화수: 46 추천대상 영화수: 9696 전체 영화수: 9742
```

6. 개인화 영화 추천 시스템 구축

- 높은 예측 평점을 가진 순으로 영화를 추천하는 함수

```
def recomm_movie_by_surprise(algo, userId, unseen_movies, top_n=10):
    # 알고리즘 객체의 predict() 메서드를 평점이 없는 영화에 반복 수행한 후 결과를 list 객체로 저장
    predictions = [algo.predict(str(userId), str(movieId)) for movieId in unseen_movies]

    # predictions list 객체는 surprise의 Predictions 객체를 원소로 가지고 있음.
    # [Prediction(uid='9', iid='1', est=3.69), Prediction(uid='9', iid='2', est=2.98),,,,]
    # 이를 est 값으로 정렬하기 위해서 아래의 sortkey_est 함수를 정의함.
    # sortkey_est 함수는 list 객체의 sort() 함수의 키 값으로 사용되어 정렬 수행.
    def sortkey_est(pred):
        return pred.est

    # sortkey_est( ) 반환값의 내림 차순으로 정렬 수행하고 top_n개의 최상위 값 추출.
    predictions.sort(key=sortkey_est, reverse=True)
    top_predictions= predictions[:top_n]

    # top_n으로 추출된 영화의 정보 추출. 영화 아이디, 추천 예상 평점, 제목 추출
    top_movie_ids = [ int(pred.iid) for pred in top_predictions]
    top_movie_rating = [ pred.est for pred in top_predictions]
    top_movie_titles = movies[movies.movieId.isin(top_movie_ids)]['title']
    top_movie_preds = [ (id, title, rating) for id, title, rating in zip(top_movie_ids, top_movie_titles, top_movie_rating)]

    return top_movie_preds
```

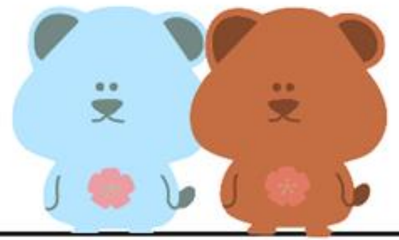
6. 개인화 영화 추천 시스템 구축

```
unseen_movies = get_unseen_surprise(ratings, movies, 9)
top_movie_preds = recomm_movie_by_surprise(algo, 9, unseen_movies, top_n=10)
print('##### Top-10 추천 영화 리스트 #####')

for top_movie in top_movie_preds:
    print(top_movie[1], ":", top_movie[2])
```

```
##### Top-10 추천 영화 리스트 #####
Usual Suspects, The (1995) : 4.3063021357008 14
Star Wars: Episode IV - A New Hope (1977) : 4.281663842987387
Pulp Fiction (1994) : 4.278152632122759
Silence of the Lambs, The (1991) : 4.226073566460876
Godfather, The (1972) : 4.1918097904381995
Streetcar Named Desire, A (1951) : 4.154746591122657
Star Wars: Episode V - The Empire Strikes Back (1980) : 4.122016128534504
Star Wars: Episode VI - Return of the Jedi (1983) : 4.108009609093436
Goodfellas (1990) : 4.083464936588478
Glory (1989) : 4.07887165526957
```

THANK YOU



EWHA EURON