Recommender Systems 1팀

하계 컨퍼런스 중간 발표

2022.07.26

김병준 김정우 김찬호 박정현

목차

1. 팀원 소개

김병준 | 컴퓨터공학

김정우 | 경영학

김찬호 | 응용통계학

박정현 | 응용통계학

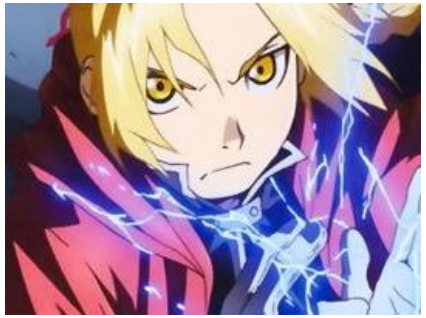
2. 주제 설명

차애캐 & 애니메이션 추천

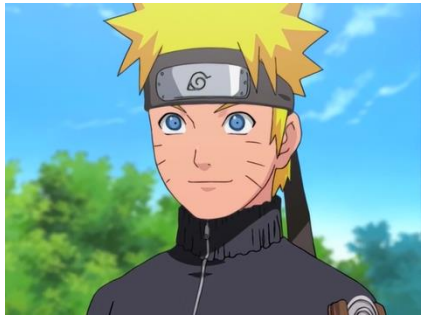
"차애캐 추천과 그에 따른 애니메이션 추천"

'최애캐'의 외모와 성격 등 다양한 요소를 종합적으로 분석해 '차애캐'를 추천하는 방법론

이어서 차애캐가 등장하는 애니메이션 소개 후 해당 애니메이션과 유사한 애니메이션 추천



금발
청소년
성급함
⋮



3. 진행 상황

데이터 전처리

1. character data에 MAL_ID(MyAnimelist_ID)추가

animation data와 character data를 연결시킬 수 있는 index필요

=> 데이터 크롤링을 통해 character data에 해당 캐릭터가 출연하는 animation의 MAL_ID추가

* character data

	Unnamed: 0	ID	Name	Alias	Gender	Hair Color	Love Rank	Hate Rank	Eye color	Birthday	Blood Type	Tags	Love Count	Hate Count	Description	url	Counts
0	0	0	L	Ryuzaki	Male	Black	1.0	48.0	Black	October 31, 1979	Unknown	Analytical, Barefoot, Detectives, Eye Bags, Sw...	44829	3447	Secretive, meticulous and cunning, L's desire ...	https://www.anime-planet.com/characters/l-deat...	48276.0

MAL_ID 추가

	Unnamed: 0	ID	Name	Alias	Gender	Hair Color	Love Rank	Hate Rank	Eye color	Birthday	Blood Type	Tags	Love Count	Hate Count	Description	url	Counts	MAL_ID
0	0	0	L	Ryuzaki	Male	Black	1.0	48.0	Black	October 31, 1979	Unknown	Analytical, Barefoot, Detectives, Eye Bags, Sw...	44829	3447	Secretive, meticulous and cunning, L's desire ...	https://www.anime-planet.com/characters/l-deat...	48276.0	1535.0

* animation data

	MAL_ID	Name	Score	Genres	English name	Japanese name	Type	Episodes	Aired	Premiered	...	Score-10	Score-9	Score-8	Score-7	Score-6	Score-5	Score-4	Score-3	Score-2	Score-1
1393	1535	Death Note	.63	Mystery, Police, Psychological, Supernatural, ...	Death Note	デスノート	TV	37	Oct 4, 2006 to Jun 27, 2007	Fall 2006	...	557406.0	535252.0	415890.0	201522.0	68577.0	28048.0	10462.0	3692.0	2256.0	3586.0

데이터 전처리

2. 불필요한 columns drop

* character data

	Unnamed: 0	ID	Name	Alias	Gender	Hair Color	Love Rank	Hate Rank	Eye color	Birthday	Blood Type	Tags	Love Count	Hate Count	Description	url	Counts	MAL_ID
0	0	0	L	Ryuzaki	Male	Black	1.0	48.0	Black	October 31, 1979	Unknown	Analytical, Barefoot, Detectives, Eye Bags, Sw...	44829	3447	Secretive, meticulous and cunning, L's desire ...	https://www.anime-planet.com/characters/l-deat...	48276.0	1535.0

	ID	CharName	Gender	HairColor	Tags	Description	url	MAL_ID
0	0	L	Male	Black	Analytical,Barefoot,Detectives,EyeBags,SweetTooth	Secretive, meticulous and cunning, L's desire ...	https://www.anime-planet.com/characters/l-deat...	1535.0

* animation data

	MAL_ID	Name	Score	Genres	English name	Japanese name	Type	Episodes	Aired	Premiered	...	Score-10	Score-9	Score-8	Score-7	Score-6	Score-5	Score-4	Score-3	Score-2	Score-1
0	1	Cowboy Bebop	8.78	Action, Adventure, Comedy, Drama, Sci-Fi, Space	Cowboy Bebop	カウボーイビバップ	TV	26	Apr 3, 1998 to Apr 24, 1999	Spring 1998	...	229170.0	182126.0	131625.0	62330.0	20688.0	8904.0	3184.0	1357.0	741.0	1580.0

	MAL_ID	Name	Genres	synopsis	Type	Source	Year
0	1	Cowboy Bebop	Action,Adventure,Comedy,Drama,Sci-Fi,Space	In the year 2071, humanity has colonized sever...	TV	Original	1990.0

모델링

1. character data를 이용한 차애캐 추천(contents-based filtering)

description 제외한 나머지 정보를 가지고 모델링

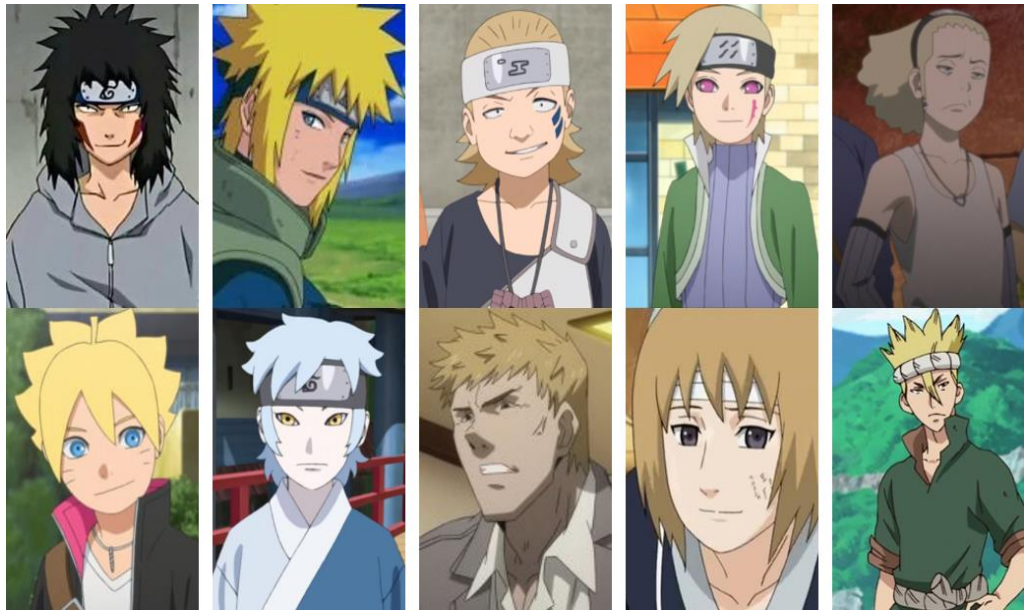
VS

description만을 사용하여 모델링

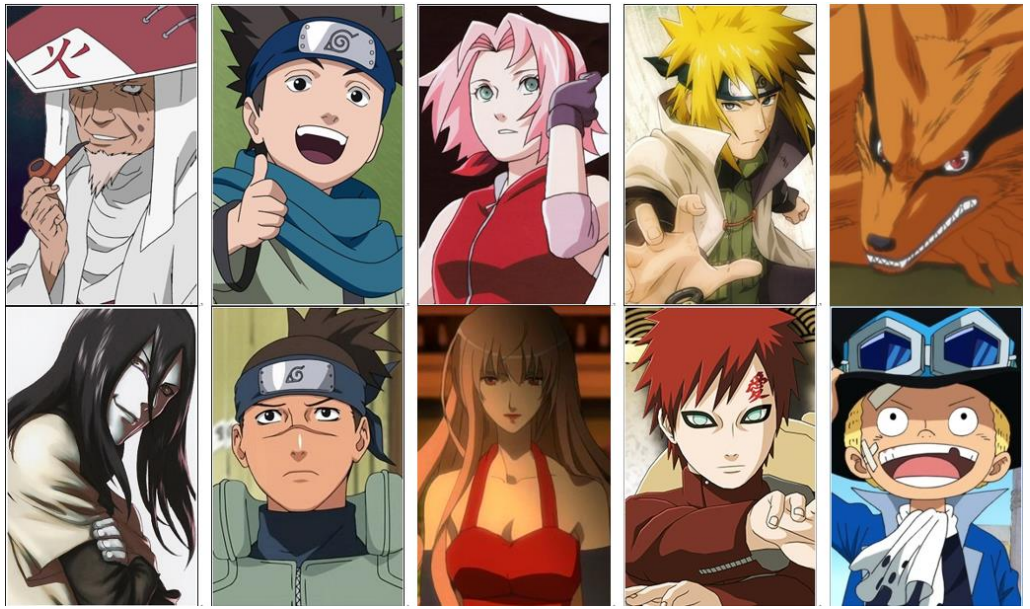
character data를 이용한 차애캐 추천 결과

Naruto UZUMAKI

description 제외



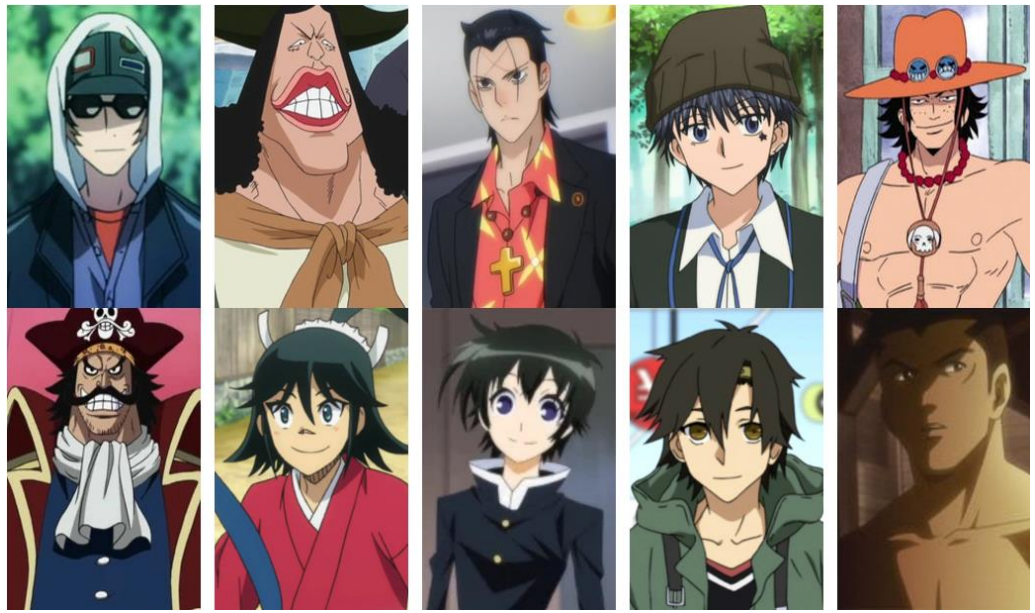
description만 사용



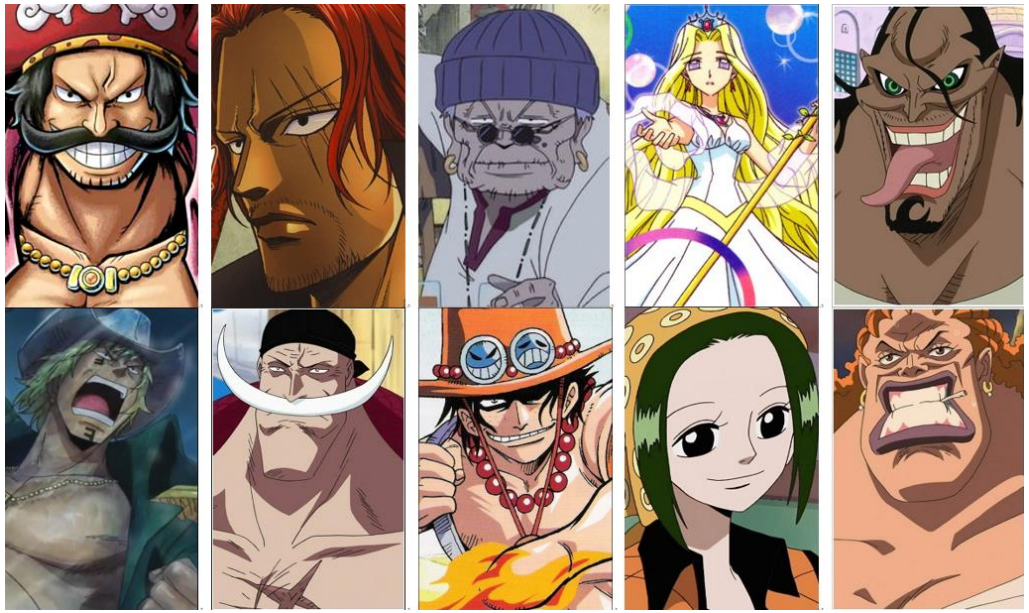
character data를 이용한 차애캐 추천 결과

Monkey D. Luffy

description 제외



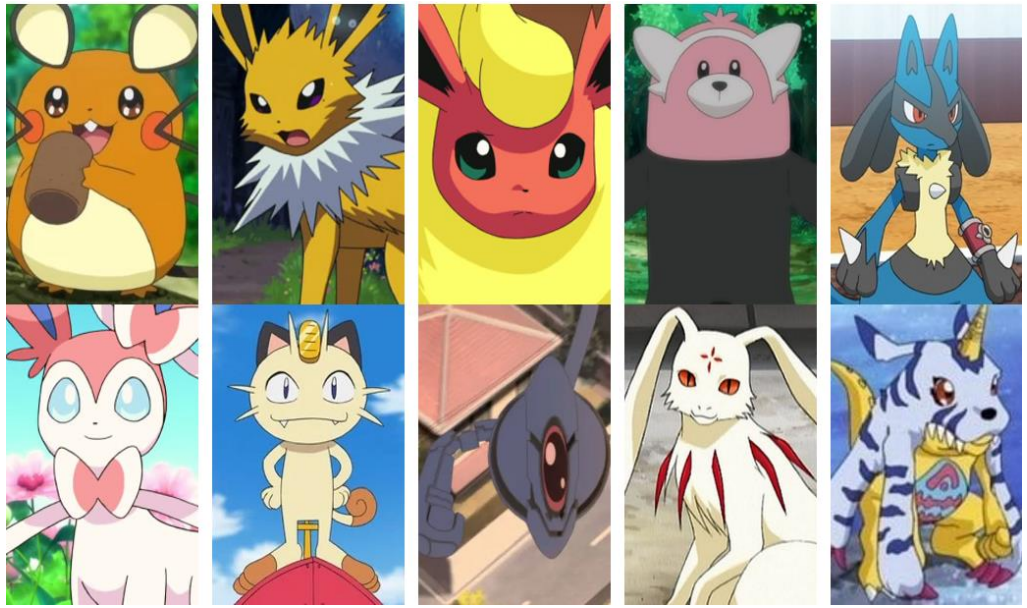
description만 사용



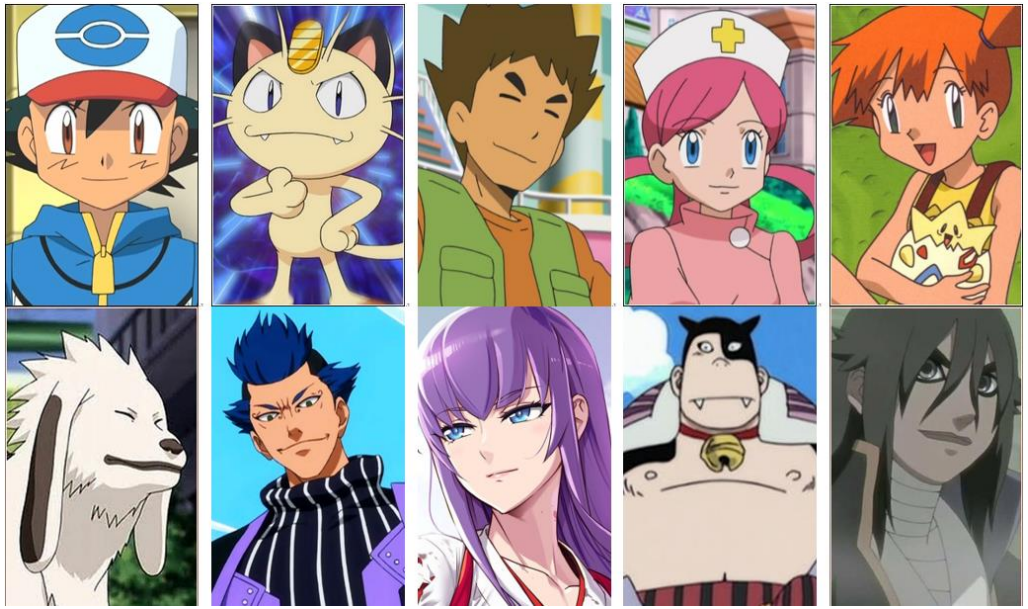
character data를 이용한 차애캐 추천 결과

Ash's Pikachu

description 제외



description만 사용

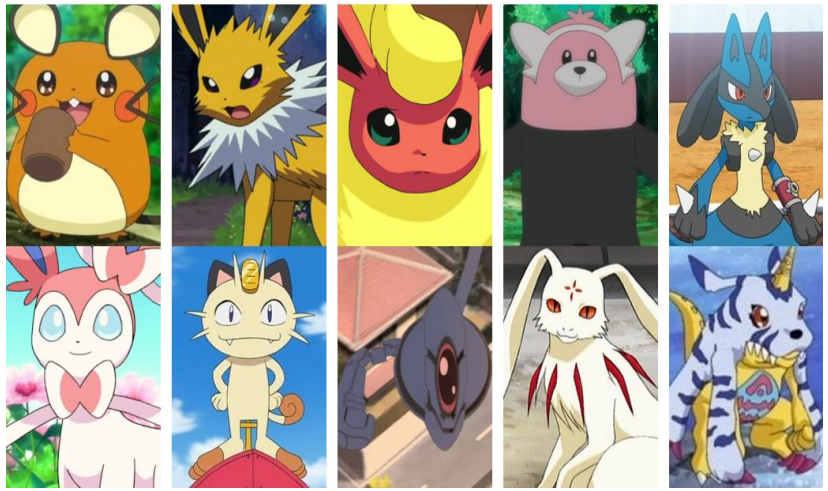


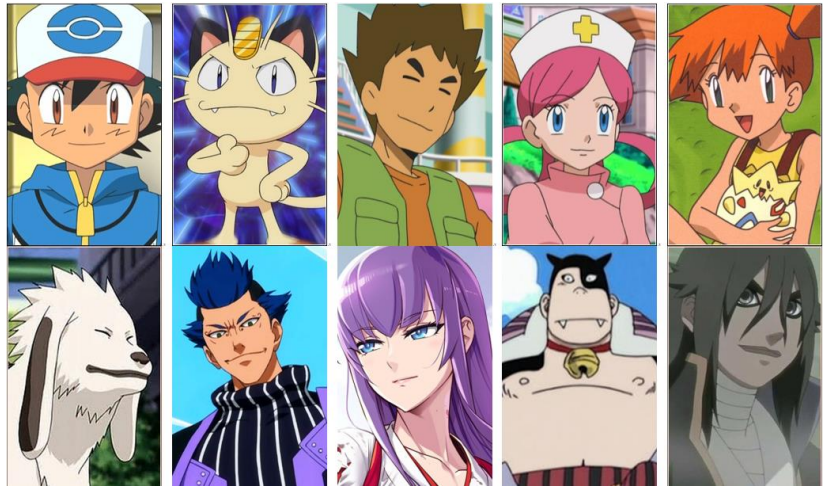
character data를 이용한 차애캐 추천 문제점

동일한 애니메이션 속의 캐릭터를 위주로 추천하는

문제 발생

(1) 아마 검색자가 이미 그 캐릭터들을 잘 알고 있을 것임
(2) 같은 애니메이션에 나왔다고 해서 비슷한 캐릭터라는 보장 어려움





character data를 이용한 차애캐 추천 문제점 - 예상 원인 및 해결 방안

* 예상 원인

description 제외

Tag의 데이터 비중이 높음

=> 같은 애니메이션에 등장하는 캐릭터일수록 tag의 단어가 같을 확률 높음

description만 사용

Description에 대한 텍스트 마이닝시, tf-idf를 이용하는데 단어의 빈도를 우선으로 삼음

=> 같은 애니메이션 캐릭터의 description이면 해당 애니메이션에서만 사용되는 고유명사를 공유

* 해결 방안

(1) 외모, 성격 등 다양한 요소를 종합적으로 고려
(2) 어떤 element에 어느 정도 weight를 줄 지 반복적으로 실험하면서 결과를 비교

모델링

2. animation data를 이용한 애니메이션 추천(contents-based filtering)

모든 정보를 가지고 모델링

VS

Genre만을 사용하여 모델링

animation data를 이용한 애니메이션 추천 결과

모두 포함

```
ani_rec_genre(anime_data, cosine_similar_full_df, 'Naruto')
```

	Name	MAL_ID
4352	Medaka Box	11761
131	Shaman King	154
8010	Boruto: Naruto Next Generations	34566
3212	Dragon Ball Z: Atsumare! Gokuu World	6714
203	Rekka no Honoo	238
5518	Dragon Ball Kai (2014)	22777
8268	Boruto: Jump Festa 2016 Special	35072
1274	Katekyo Hitman Reborn!	1604
1366	Naruto: Shippuuden	1735
1178	D.Gray-man	1482
3035	Dragon Ball Kai	6033
233	Bleach	269
684	Dragon Ball Z	813
428	Ueki no Housoku	479
1329	Duel Masters	1685
11700	Shaman King (2021)	42205
824	Dragon Ball GT: Gokuu Gaiden! Yuuki no Akashi ...	987
6457	Dragon Ball Super	30694
762	Dragon Ball Z Movie 11: Super Senshi Gekiha!! ...	904
189	Dragon Ball	223

조금 더 다채로운 결과

Genre만 사용

```
ani_rec_genre(anime_data, cosine_similar_df, 'Naruto')
```

	Name	MAL_ID
4611	Naruto: Shippuuden Movie 6 - Road to Ninja	13667
6980	Boruto: Naruto the Movie - Naruto ga Hokage ni...	32365
5509	Dragon Ball Z: Summer Vacation Special	22695
203	Rekka no Honoo	238
3212	Dragon Ball Z: Atsumare! Gokuu World	6714
8268	Boruto: Jump Festa 2016 Special	35072
5518	Dragon Ball Kai (2014)	22777
4176	Naruto Soyokazeden Movie: Naruto to Mashin to ...	10659
533	Naruto: Takigakure no Shitou - Ore ga Eiyuu Da...	594
4181	Naruto: Honoo no Chuunin Shiken! Naruto vs. Ko...	10686
1366	Naruto: Shippuuden	1735
3035	Dragon Ball Kai	6033
3114	Naruto: Shippuuden Movie 3 - Hi no Ishi wo Tsu...	6325
684	Dragon Ball Z	813
5806	Dragon Ball Z Movie 15: Fukkatsu no "F"	25389
6069	Boruto: Naruto the Movie	28755
3574	Naruto: Shippuuden Movie 4 - The Lost Tower	8246
824	Dragon Ball GT: Gokuu Gaiden! Yuuki no Akashi ...	987
6457	Dragon Ball Super	30694
762	Dragon Ball Z Movie 11: Super Senshi Gekiha!! ...	904

유사한 결과 얻음

animation data를 이용한 애니메이션 추천 - 예상 원인 및 해결 방안

데이터의 각 종류를 하나의 새로운 속성으로 변경하여 유사도행렬을 만들 때,
장르가 가지는 영향력이 너무 클 것으로 생각됨

* 예상 원인

장르는 총 42개로 많은 종류의 데이터를 가지는 반면
나머지 데이터는 종류가 많지 않음

* 해결 방안

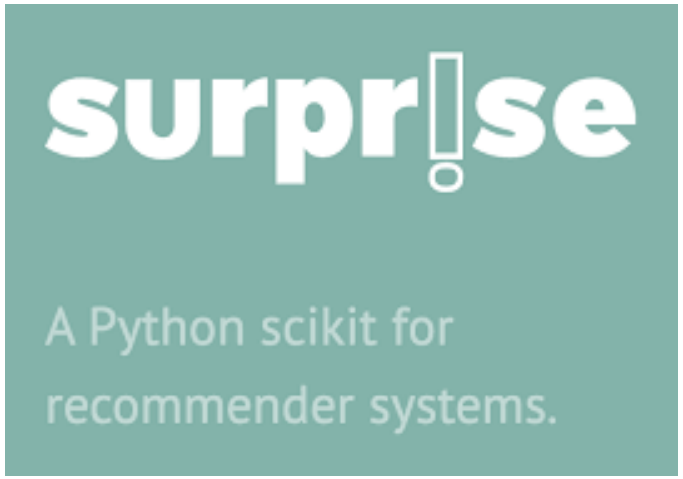
단순히 0,1 원핫인코딩 형식이 아니라 가중치를 변경해가며 유사도행렬을 만든다면
더 좋은 결과를 얻을 수 있을 것으로 예상

모델링

3. animation data를 이용한 애니메이션 추천(collaborative Filtering)

"나와 성향이 비슷한 친구들은 어떤 애니메이션을 찾아볼까?"

=>많은 사용자들로부터 얻은 기호정보(taste information)에 따라 사용자들의 관심사들을 자동적으로 예측하게 해주는 방법



animation data를 이용한 애니메이션 추천 - 데이터셋

	user_id	anime_id	rating
0	0	430	9
1	0	1004	5
2	0	3010	7
3	0	570	7
4	0	2762	9

Surprise를 활용하여 추천시스템을 구현하고자 할 때,
‘사용자 - 아이템 - 평점’으로 데이터 가공 필요

animation data를 이용한 애니메이션 추천 - 전처리

```
min_anime_ratings = 5
filter_animes = ratings['anime_id'].value_counts() >= min_anime_ratings
filter_animes = filter_animes[filter_animes].index.tolist()

ratings_new = ratings[ratings['anime_id'].isin(filter_animes)]
print('The original data frame shape:₩t{}'.format(ratings.shape))
print('The new data frame shape:₩t{}'.format(ratings_new.shape))
```

```
The original data frame shape:  (57633278, 3)
The new data frame shape:       (57632002, 3)
```

- 5명 미만의 user에게 평가를 받은 애니메이션 제거

```
min_user_ratings = 5
filter_users = ratings_new['user_id'].value_counts() >= min_user_ratings
filter_users = filter_users[filter_users].index.tolist()

ratings_new2 = ratings_new[ratings_new['user_id'].isin(filter_users)]
print('The original data frame shape:₩t{}'.format(ratings_new.shape))
print('The new data frame shape:₩t{}'.format(ratings_new2.shape))
```

```
The original data frame shape:  (57632002, 3)
The new data frame shape:       (57596151, 3)
```

- 5개 미만의 애니메이션을 평가한 user 제거

```
for a in range(len(ratings_new2)):
    if ratings_new2["anime_id"][a] not in list(anime["MAL_ID"][:]):
        ratings_new2.drop(index=a)
```

- anime 데이터셋에 있는 애니메이션을 제외하고 drop

animation data를 이용한 애니메이션 추천 - 알고리즘

random_pred.NormalPredictor	Algorithm predicting a random rating based on the distribution of the training set, which is assumed to be normal.
baseline_only.BaselineOnly	Algorithm predicting the baseline estimate for given user and item.
knns.KNNBasic	A basic collaborative filtering algorithm.
knns.KNNWithMeans	A basic collaborative filtering algorithm, taking into account the mean ratings of each user.
knns.KNNWithZScore	A basic collaborative filtering algorithm, taking into account
knns.KNNBaseline	A basic collaborative filtering algorithm taking into account a *baseline* rating.
matrix_factorization.SVD	The famous *SVD* algorithm, as popularized by Simon Funk during the Netflix Prize.When baselines are not used, this is equivalent to Probabilistic Matrix Factorization.
matrix_factorization.SVDpp	The *SVD++* algorithm, an extension of SVD taking into account implicit ratings.
matrix_factorization.NMF	A collaborative filtering algorithm based on Non-negative Matrix Factorization.
slope_one.SlopeOne	A simple yet accurate collaborative filtering algorithm.
co_clustering.CoClustering	A collaborative filtering algorithm based on co-clustering.

animation data를 이용한 애니메이션 추천 - cross validation

```
benchmark = []
# 모든 알고리즘을 literate화 시켜서 반복문을 실행시킨다.
for algorithm in [SVD(), SVDpp(), SlopeOne(), NMF(), KNNBaseline(), KNNBasic(), KNNWithMeans(), KNNWithZScore(), BaselineOnly(), CoClus

    # 교차검증을 수행하는 단계.
    results = cross_validate(algorithm, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)

    # 결과 저장과 알고리즘 이름 추가.
    tmp = pd.DataFrame.from_dict(results).mean(axis=0)
    tmp = tmp.append(pd.Series([str(algorithm).split(' ')[0].split('.')[-1]], index=['Algorithm']))
    benchmark.append(tmp)

pd.DataFrame(benchmark).set_index('Algorithm').sort_values('test_rmse')
```

animation data를 이용한 애니메이션 추천 - Grid Search

KNN 기반 알고리즘(아이템 기반 최근점 이웃 CF)의 파라미터

- **k**: 이웃의 크기
- **name**: 사용할 유사도의 종류를 나타내는 문자열. 디폴트는 'MSD'.
  - 평균제곱차이 유사도 (Mean Squared Difference Similarity)
  - 코사인 유사도 (Cosine Similarity)
  - 피어슨 유사도 (Pearson Similarity)
  - 피어슨-베이스라인 유사도 (Pearson-Baseline Similarity)
- **user_based**: True면 사용자 기반, False면 상품 기반.

SVD 기반 알고리즘(행렬 분해를 이용한 잠재요인 CF)의 파라미터

- **n_epochs**: SGD의 반복 횟수 지정
- **n_factors**: SVD의 잠재 요인 K의 크기 지정
- **lr_all**: 학습률
- **reg_all**: 규제 텀 계수

animation data를 이용한 애니메이션 추천 - 추천시스템 구현

```python
def recomm_anime_by_surprise(algo, user_id, unseen_animes, top_n=10):
    # 알고리즘 객체의 predict() 메서드를 평점이 없는 애니메이션에 반복 수행한 후 결과를 list 객체로 저장
    predictions = [algo.predict(str(user_id), str(anime_id) for anime_id in unseen_animes]

    # 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 애니메이션의 정보 추출. 애니메이션 아이디, 추천 예상 평점, 제목 추출
    top_anime_ids = [ int(pred.iid) for pred in top_predictions]
    top_anime_rating = [pred.est for pred in top_predictions]
    top_anime_names = anime[anime.MAL_ID.isin(top_anime_ids)]['title']
    top_anime_preds = [ (id, title, rating) for id, title, rating in zip(top_anime_ids, top_anime_names, top_anime_rating)]

    return top_anime_preds

unseen_animes = get_unseen_animelist(9)
top_anime_preds = recomm_anime_by_surprise(algo, 9, unseen_animes, top_n=10)
print('##### Top-10 추천 애니메이션 리스트 #####')

for top_anime in top_anime_preds:
    print(top_anime[1], ":", top_anime[2])
```

1. **사용할 알고리즘과 추천 대상자인 User의 id를 입력받음.**
2. **User_id에 해당하는 사용자가 이미 평점을 매긴, 즉 이미 시청한 애니메이션을 제외.**
3. **입력된 알고리즘에 의해 사용자가 평점을 매기지 않은 애니메이션의 평점을 예측한 후,**
4. **예측된 평점 중 가장 높은 평점을 받은 상위 N개의 애니메이션을 추천**

4. 향후 계획

8월

week1
[모델링 완성]
week2
[예선 제출 마감]
week3
[본선 발표 준비 및 발표]

감사합니다