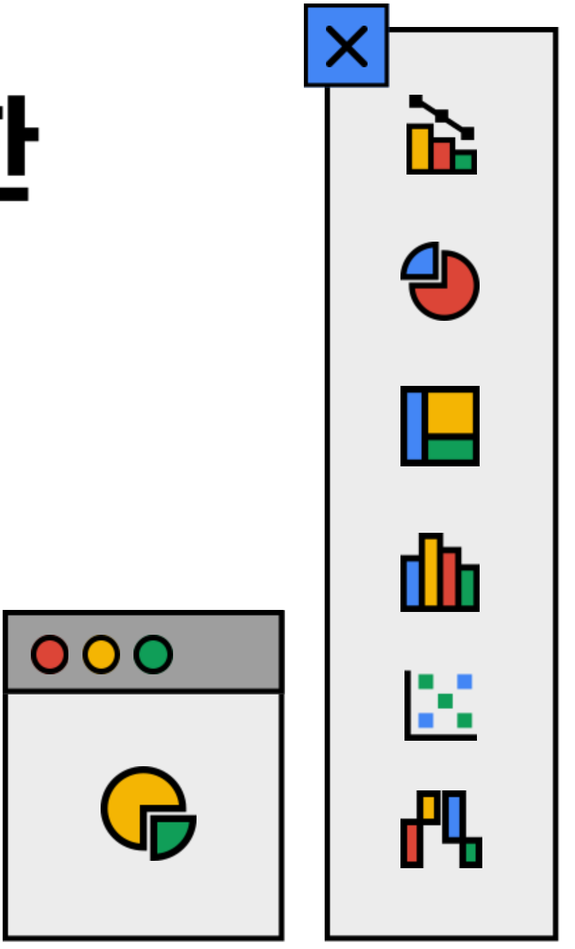
Amazon Review Data를 활용한

추천시스템 구현 및 분석

24-2 빅데이터 분석 및 활용 기말프로젝트

정보통계학전공 이수민



CONTENTS

Amazon Review Data를 활용한 추천시스템 구현 및 분석

서론

추천시스템이란?

사용자의 정보 데이터를 분석하여 개인의 취향에 맞는 아이템을 추천하는 알고리즘

amazon.com NETFLIX

YouTube Spotify®

서론

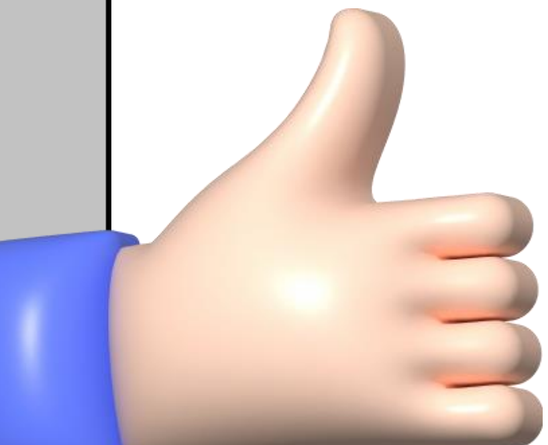
추천시스템이란?

사용자의 리뷰 및 평점 데이터를 기반으로 한 추천 시스템은 제품 탐색 시간을 줄이고 사용자 만족도를 높이는 데 중요한 역할을 함.
Amazon과 같은 대형 온라인 플랫폼은 방대한 user-item 상호작용 데이터를 보유하고 있으며,
이를 활용해 각 사용자별 개인화된 추천을 제공하고 있음

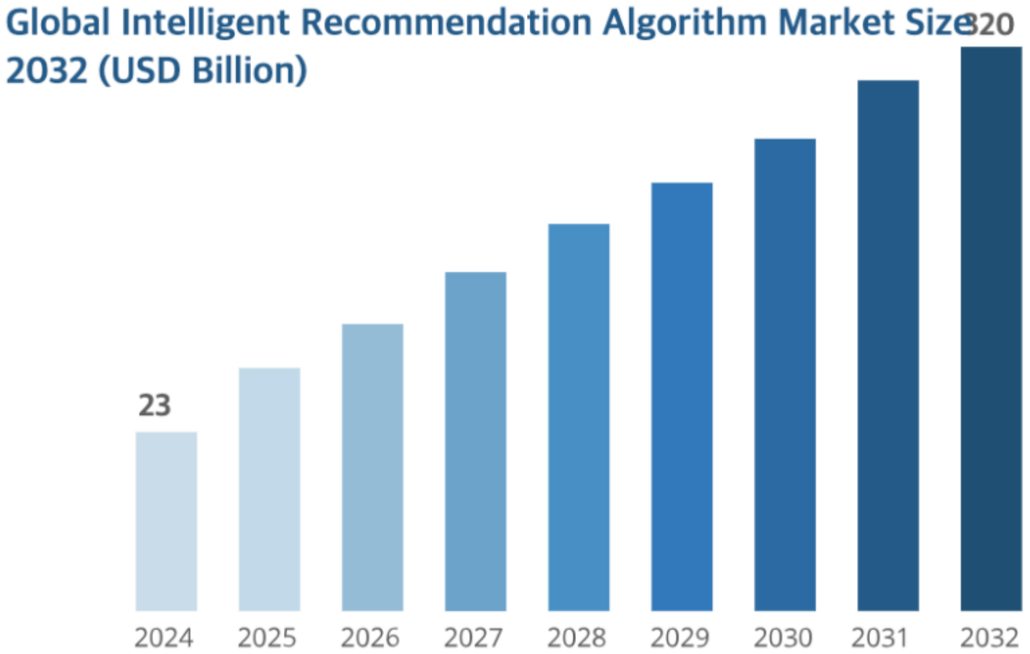
'무신사'와 '네이버 쇼핑'으로 보는 개인화 추천 시스템의 성공

OTT 넷플릭스가 촉발한 취향저격 '추천 알고리즘'

" 추천 엔진 시장은 IT 및 통신 부문에서 유망한 속도로 성장하고 있습니다 "

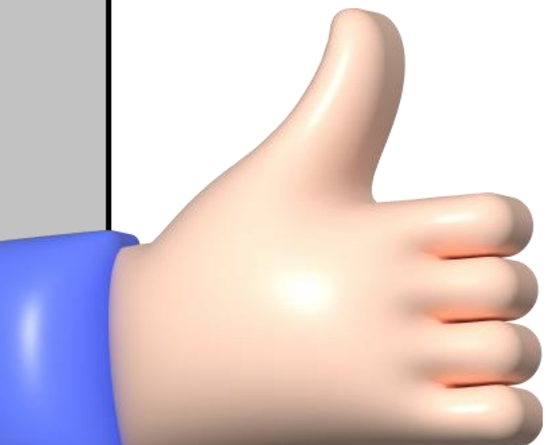


서론



점점 증가할 것으로 보이는 추천 시스템 시장 성장세

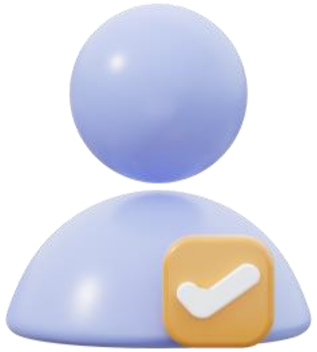
출처 : business research insights



서론

프로젝트 목적

1. 추천 시스템의 Baseline 모델을 구현
2. LightGCN과 같은 다양한 최신 모델을 활용해 추천 시스템의 성능을 개선.
3. 다양한 추천 모델을 비교 분석하여 정확도와 효율성을 평가.
4. 사용자와 제품의 특성을 심층적으로 분석함으로써 추천 시스템의 실제 비즈니스 환경에서의 활용 가능성과 확장성



데이터 수집 및 전처리

01 데이터 수집

Amazon이 제공하는 리뷰 데이터셋 중
전자 제품 데이터셋을 수집해 추천시스템 분석 데이터로 활용

<Amazon Review Data : ratings_Electronics 속성>

columns	dtype	비고
userID	object	각 사용자를 고유하게 식별하는 ID
productID	object	각 상품을 고유하게 식별하는 ID
Rating	float64	해당 사용자가 상품에 대해 준 평점
Timestamp	int64	평점을 매긴 시간

데이터 수집 및 전처리

02 데이터 전처리

1) 누락된 값 확인

```
print('Number of missing values across
columns: \n',df.isnull().sum())
```

3) 분석에 사용하지 않는 필요없는 컬럼 제거

```
df.drop('timestamp'], axis=1,inplace=True)
```

3) 데이터 샘플링 (데이터 수가 커 모델이 잘 돌아가지 않는 경우를 위해 전체 데이터의 **20%만을 사용**, 추후 분석시 각 분석 방법마다 샘플링 진행 예정 → 데이터의 크기가 커 모델이 잘 돌아가지 않는다는 문제점이 있음)

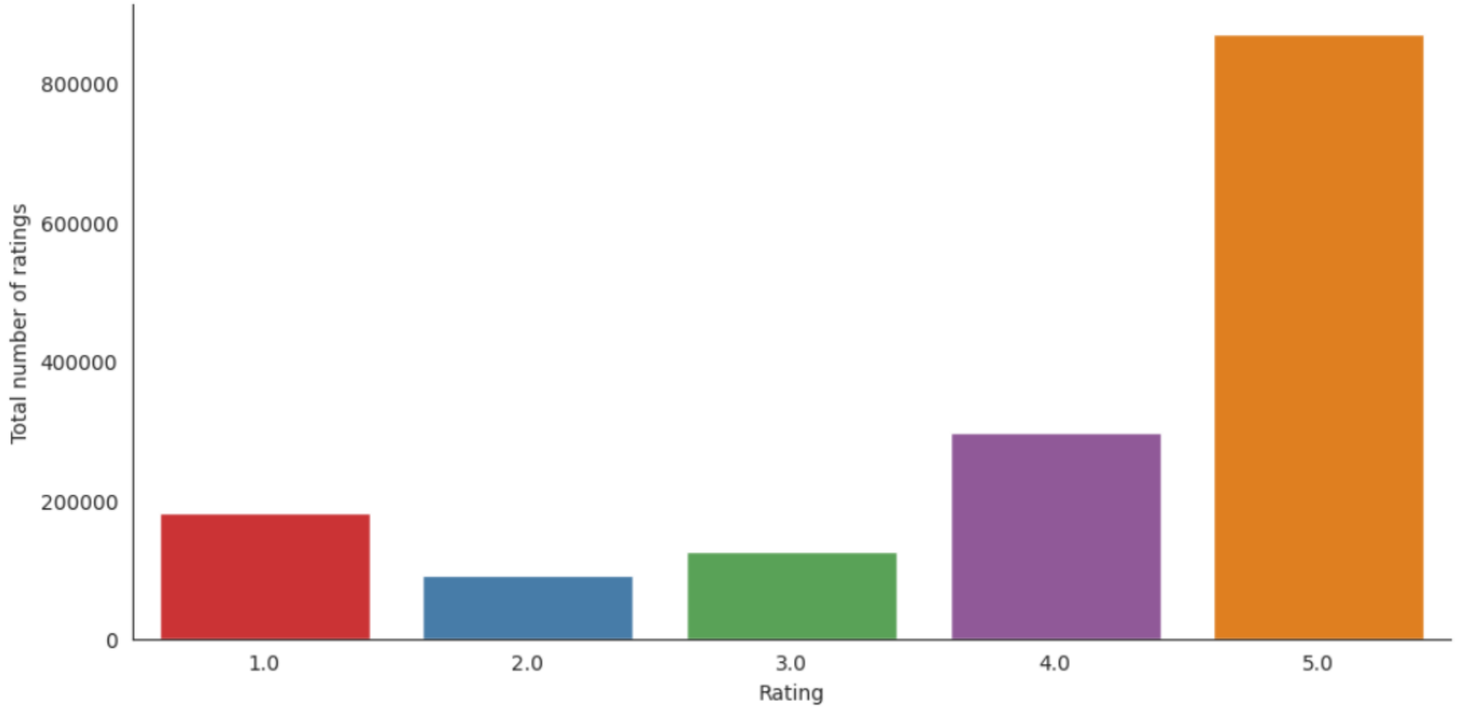
```
sampled_df = df.sample
                (frac=0.2, random_state=42)
```

```
Original Data Ratings Mean: 4.012336791112817
Sampled Data Ratings Mean: 4.011706848250618
```

원본 데이터와 샘플링 데이터의 통계 비교

데이터 EDA

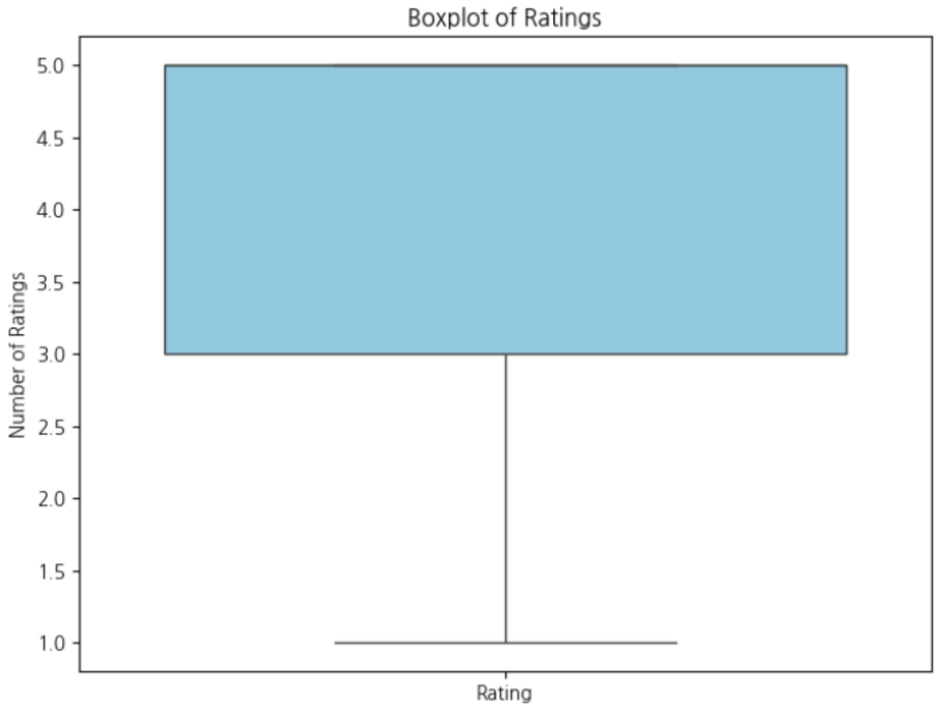
01 평점 분포 시각화



많은 사람들이 평점 5점을 준 것을 알 수 있음

데이터 EDA

01 평점 분포 시각화



대체로 사용자들의 평점은 긍정적인 것을 알 수 있음

데이터 EDA

02 사용자 및 제품 정보 확인

1) 사용자 및 제품 정보 확인

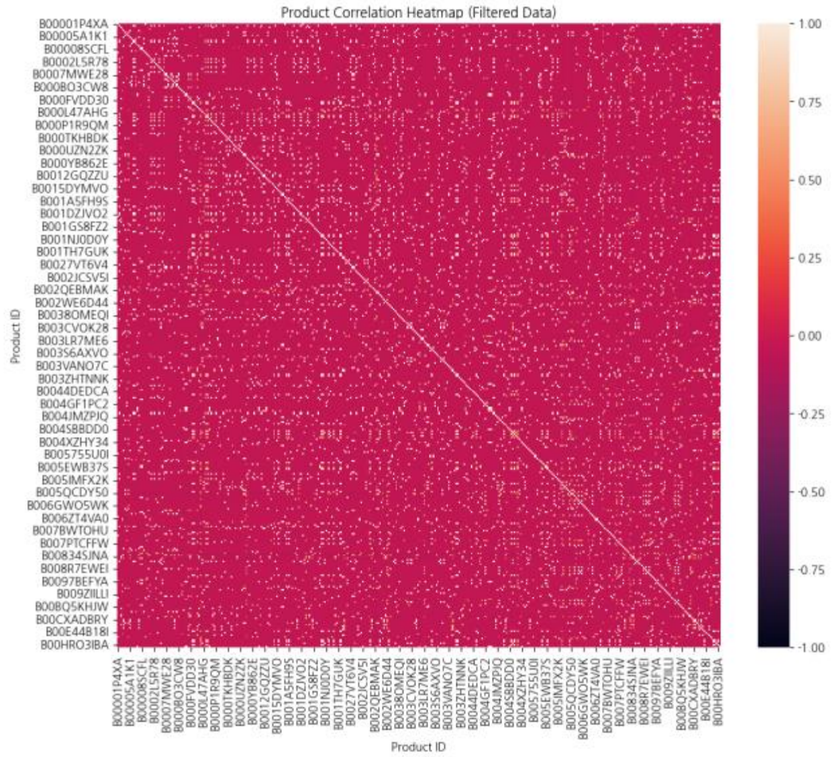
총 평점 수 : 1564896
총 사용자 수 : 1225827
총 제품 수 : 237939

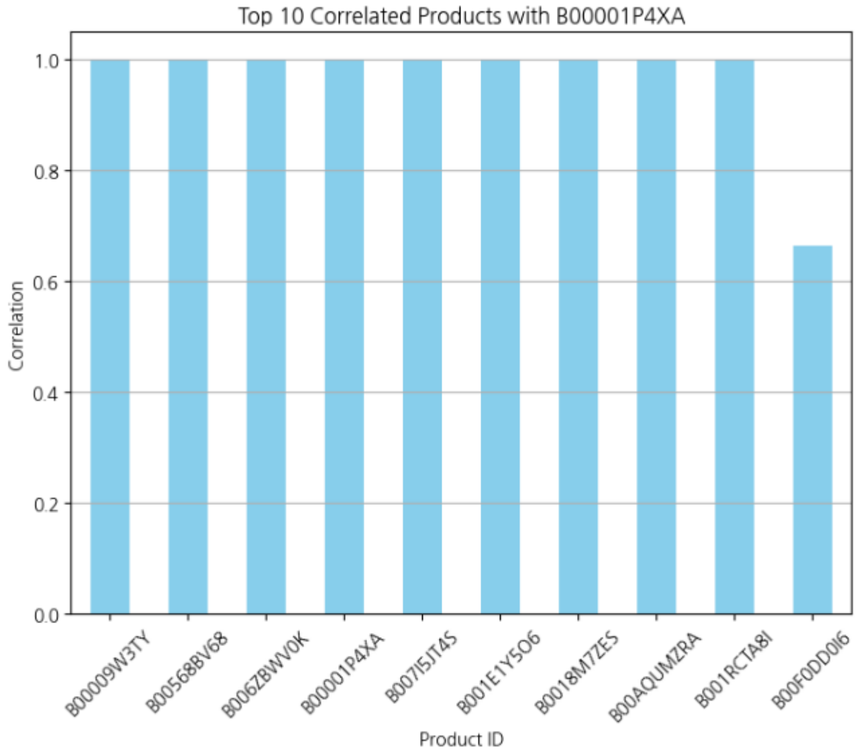
2) 사용자별 평점 수 분석

```
상위 5명 평점 수 확인
 userID
ADLVFFE4VBT8       107
A3OXHLG6DIBRW8     105
A5JLAU2ARJ0BO      102
A6FIAB28IS79        91
A1ODOGXEYECQQ8      86
Name: Rating, dtype: int64
사용자별 평점 수 통계 요약
 count    1.225827e+06
mean      1.276604e+00
std       1.009242e+00
min       1.000000e+00
25%       1.000000e+00
50%       1.000000e+00
75%       1.000000e+00
max       1.070000e+02
```

데이터 EDA

03 제품 간 상관관계 시각화

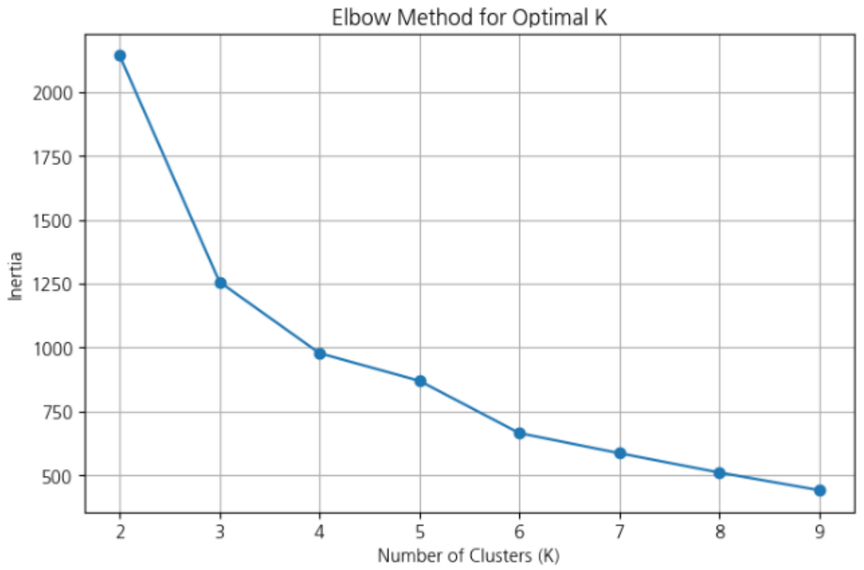


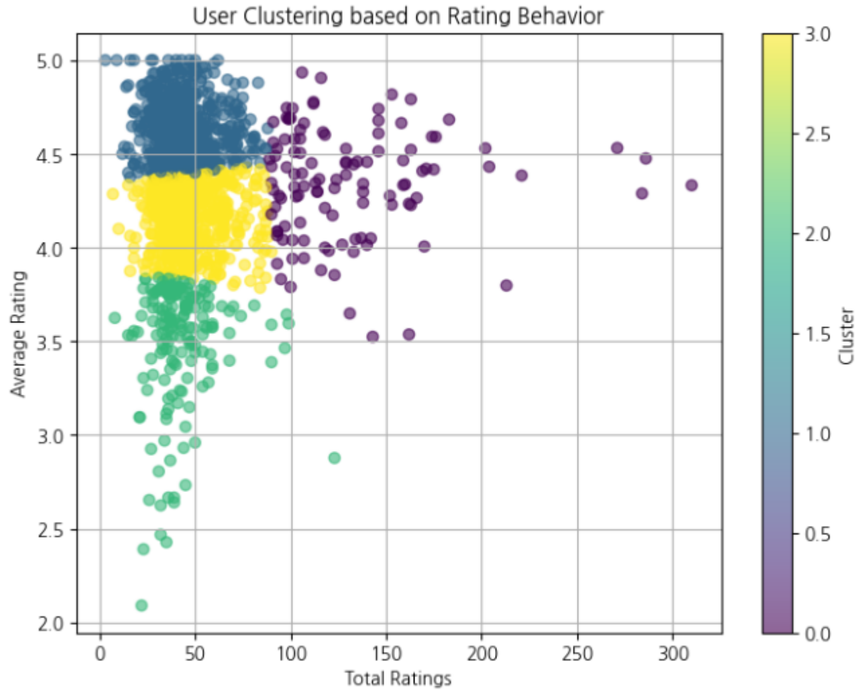


제품 간 상관관계를 구해보고 특정 제품과 연관된 상위 10개 제품 추출

데이터 EDA

04 군집 분석(사용자 그룹 클러스터링)

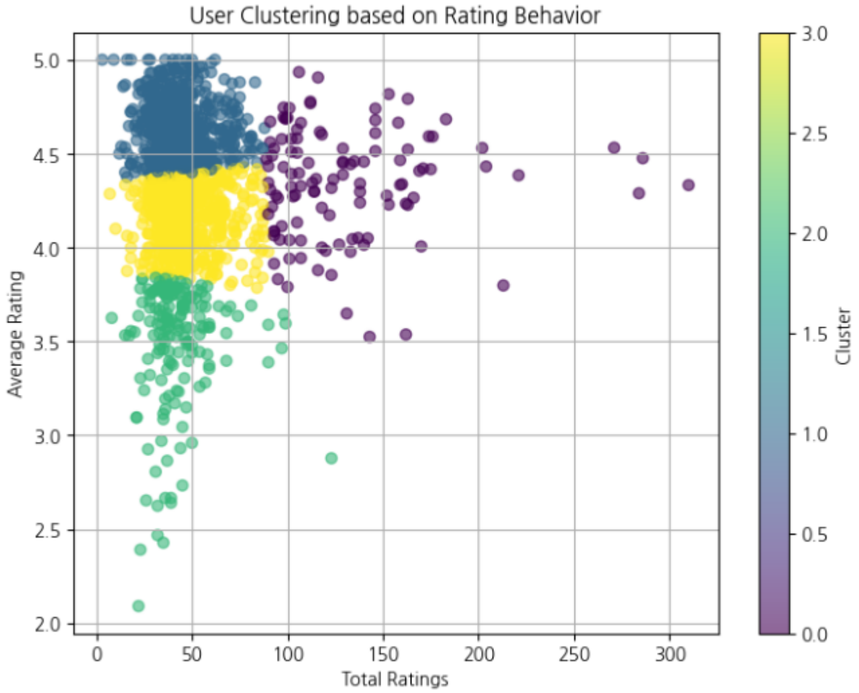




K = 4 / 사용자들의 평점 행동을 기반으로 한 클러스터링 결과

데이터 EDA

04 군집 분석(사용자 그룹 클러스터링)



- **클러스터 1과 클러스터 2 의 차이:**
둘 다 총 평점 수는 적지만, 파란색 그룹은 높은 평점을 주는 반면,
초록색 그룹은 낮은 평점을 부여
→ 이는 사용자 만족도나 성향에 따라 확연히 구분될 수 있음을 나타냄

- **클러스터 0**
이 그룹은 평점을 가장 많이 남기지만 평균 평점이 중간 수준.
→ 평점 수가 많아 다양한 상품이나 서비스에 대한 의견을 적극적으로 표현하는 사용자들

- **클러스터 3**
적당한 평점 수와 비교적 높은 평균 평점을 보임
→ 이 그룹은 활동적이면서도 긍정적인 성향을 가진 사용자로 해석

추천 시스템 구현 및 분석

01 인기 기반 추천 시스템

1. 평점 수가 50개 이상인 제품만 선택하여 분석에 사용

2. 제품별 평점 수 분석

3. 제품별 평균 평점 분석

4. 평점 수 및 평균 평점 분포 시각화

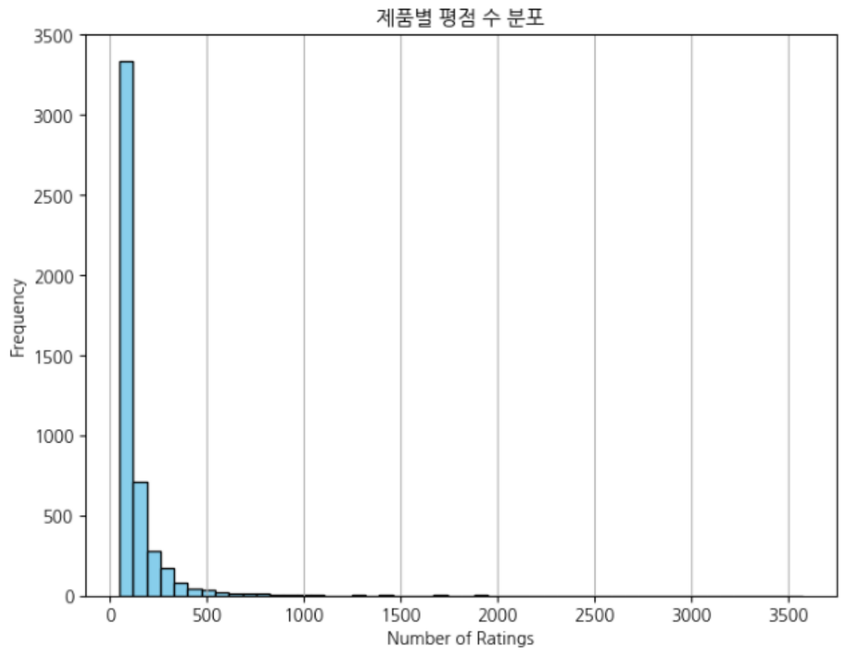
5. 평점 수와 평균 평점 관계 분석

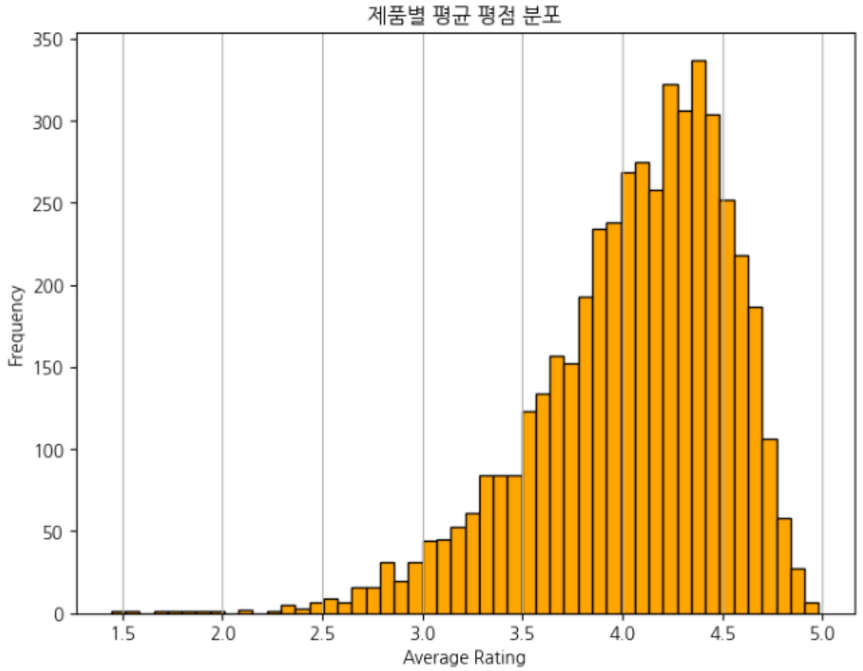
6. 상위 30개 인기 제품 시각화

```
제품별 평균 평점 (상위 5개):
productID
0972683275    4.563452
1400501466    3.581818
1400532655    3.568421
140053271X    3.925000
B00000DM9W    4.635294
Name: Rating, dtype: float64
평균 평점이 높은 상위 5개 제품:
productID
B00CG70K78    4.981481
B0033PRWSW    4.952941
B008VGCT9K    4.949153
B005LJQOPK    4.943396
B005LJQM3Y    4.926606
Name: Rating, dtype: float64
평점 수가 많은 상위 5개 제품:
productID
B0074BW614    3571
B00DR0PDNE    3326
B007WTAJTO    2873
B0019EHU8G    2398
B006GWO5WK    2350
```

추천 시스템 구현 및 분석

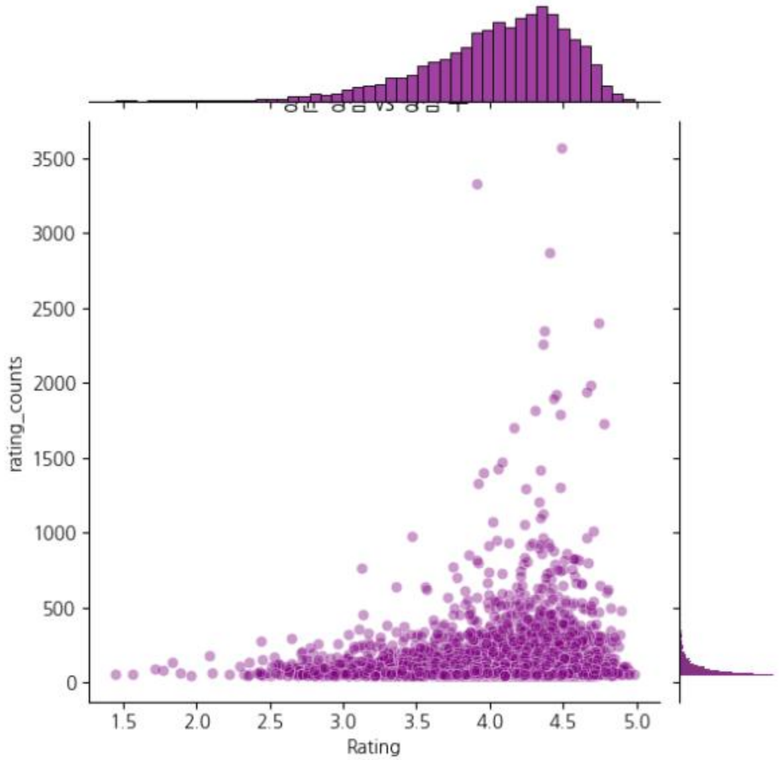
01 인기 기반 추천 시스템





추천 시스템 구현 및 분석

01 인기 기반 추천 시스템



<평점 분포 히스토그램과 평점 수 분포 히스토그램이 추가된 Jointplot>

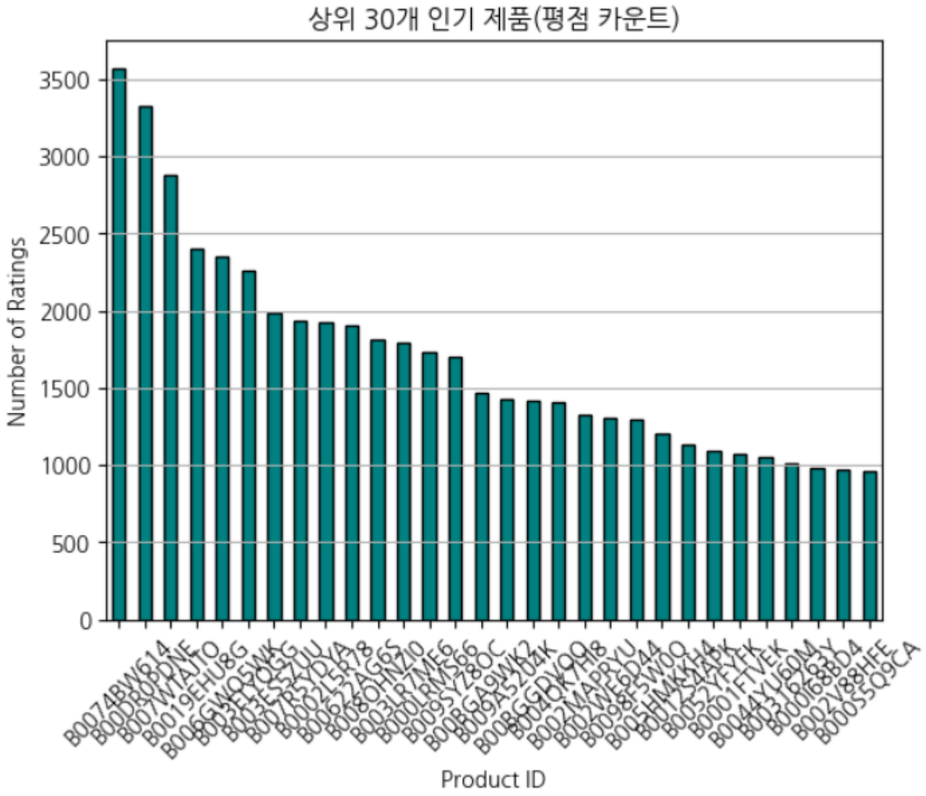
- 평점의 양극화:
사용자들은 주로 높은 평점(4.0 ~ 5.0)을 부여하며, 낮은 평점(1.5 ~ 2.5)은 드문 편, 이는 사용자들이 극단적으로 불만족하지 않는 한 평균 이상의 점수를 주는 경향이 있음

- 인기 제품의 집중화:
일부 항목은 평점 수가 압도적으로 많음 → 이들은 핵심 인기 제품으로 간주
대부분의 항목은 평점 수가 적음 → 긴 꼬리 효과가 존재

- 추천 시스템 적용:
평점 수가 많고 평점이 높은 제품을 우선 추천하는 인기 기반 추천 시스템이 효과적, 평점 수가 적은 제품에 대한 추천도 강화하려면 Collaborative Filtering과 같은 방법도 필요

추천 시스템 구현 및 분석

01 인기 기반 추천 시스템

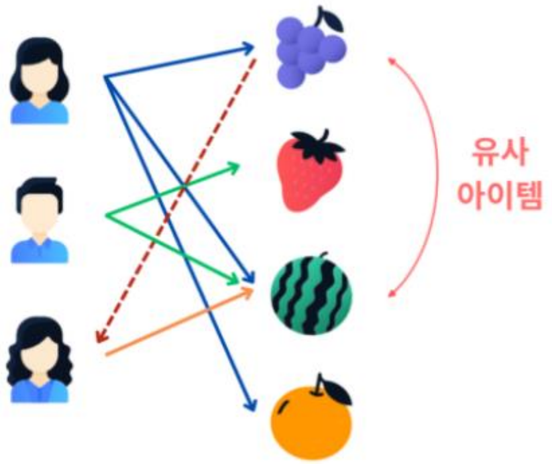


추천 시스템 구현 및 분석

02 아이템 기반 협업 필터링(Collarborative Filtering)

사용자와 아이템 간의 상호작용(평점)을 기반으로 추천을 수행
아이템 기반(Item-Item)이란 특정 아이템과 유사한 다른 아이템을 찾아 사용자에게 추천하는 알고리즘

- K-Nearest Neighbors 알고리즘을 사용 (평점 평균을 활용)
- 피어슨 상관계수와 코사인 유사도를 기반으로 유사도 계산



아이템 기반 협업 필터링

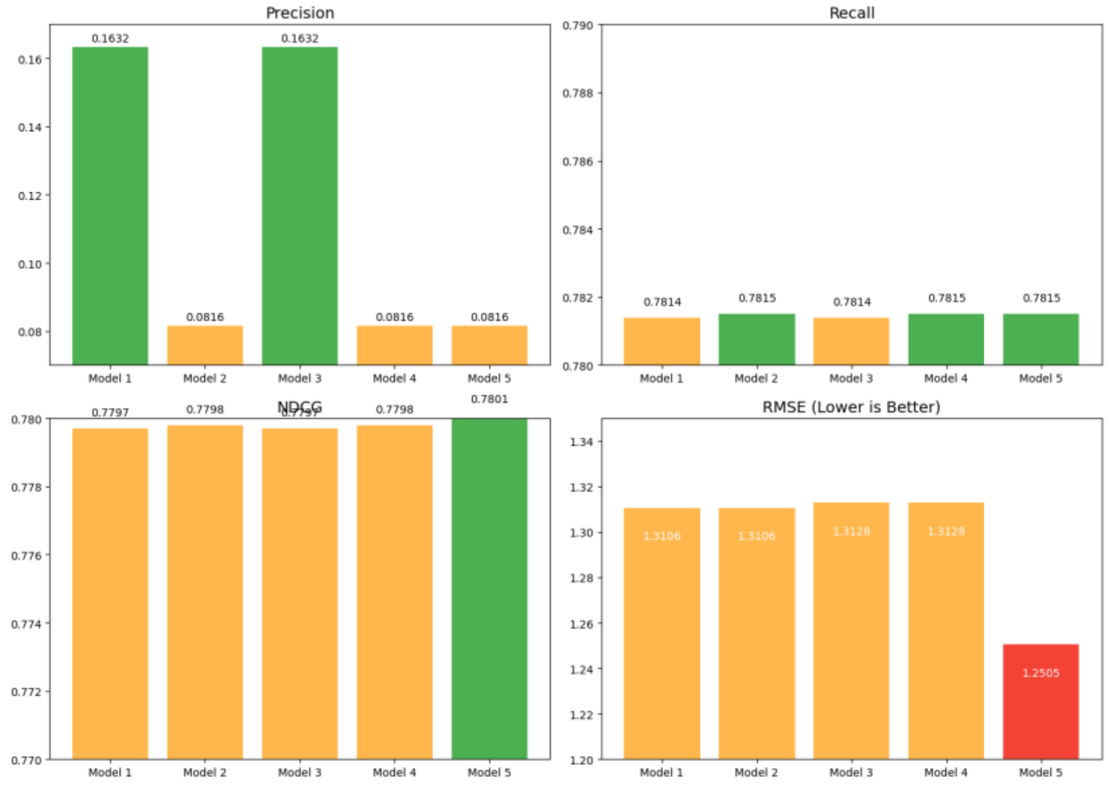
추천 시스템 구현 및 분석

02 아이템 기반 협업 필터링(Collarborative Filtering)

	Model 1	Model 2	Model 3	Model 4	Model 5
유사도	피어슨 상관계수	피어슨 상관계수	코사인 유사도	코사인 유사도	유사도 X 행렬 분해 알고리즘 사용
K	K = 5	K = 10	K = 5	K = 10	K = 10
Precision	0.1632	0.0816	0.1632	0.0816	0.0816
Recall	0.7814	0.7815	0.7814	0.7815	0.7815
NDCG	0.7797	0.7798	0.7797	0.7798	0.7801
RMSE	1.3106	1.3106	1.3128	1.3128	1.2505

추천 시스템 구현 및 분석

02 아이템 기반 협업 필터링(Collarborative Filtering)



→ Model 5는 RMSE가 가장 낮고,
NDCG도 가장 높아 전반적으로 가장 성능이 우수
Precision이 낮긴 하지만,
Recall과 NDCG가 높기 때문에
사용자의 관심 항목을 더 잘 예측하고
정렬 순서를 더 잘 맞췄다고 볼 수 있음

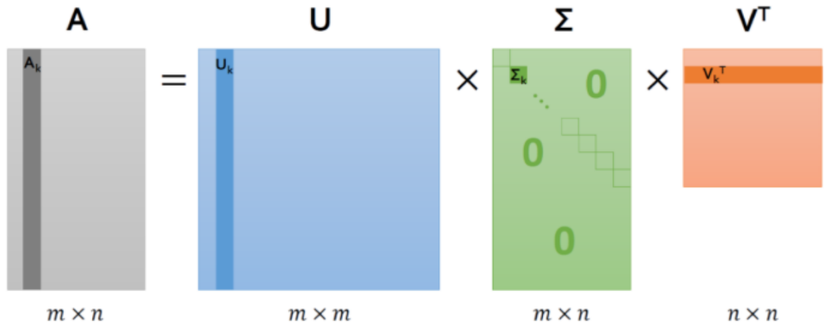
→ 코사인 유사도보다 피어슨 상관계수를 유사도로 쓴 모델
조금 더 성능이 좋다는 것을 알 수 있음

추천 시스템 구현 및 분석

03 Model-based Collaborative Filtering - MF(행렬분해)

- 상위 k개의 특이값과 벡터만 계산하고 SVD에 비해 계산량이 적은 Truncated SVD 사용

1. 사용자와 제품 간의 평점 데이터를 사용하여 사용자 - 아이템 행렬을 생성
2. 행렬을 전치 : 아이템 간 유사도를 계산하기 위해
3. 행렬 분해 : SVD를 통해 차원을 축소하여 아이템 벡터를 저차원 공간에서 표현
4. 상관 행렬 계산 : 저차원 벡터 표현을 사용하여 각 아이템 간 상관 관계를 계산
5. 유사한 아이템 찾기



추천 시스템 구현 및 분석

03 Model-based Collaborative Filtering - MF(행렬분해)

```
Correlation Matrix Shape: (3576, 3576)
Product ID at Index 75: B00006RVPW
Top 5 similar items for Product ID B00006RVPW: ['B0016P5ASK', 'B00EIQTKAS', 'B000HZGQ9C', 'B002IC0YL8', 'B0002Y5WZM']
```

<한 아이템을 고르고 상위 5개의 유사한 아이템 찾기>

	MF
K	K = 10
Precision	0.0777
Recall	0.7742
NDCG	0.7742
RMSE	4.3219

정확도가 낮고 예측된 평점과 실제 평점 사이의 오차가 큰 것을 알 수 있음

추천 시스템 구현 및 분석

03 Model-based Collaborative Filtering - MF(행렬분해)

- 장점
  - 효율성 : Truncated SVD는 데이터 희소성을 처리하며 대규모 데이터셋에서도 효과적으로 동작
  - 일반화 : 기존의 평점 데이터만으로 새로운 사용자/아이템에 대해 추천할 수 있음
- 단점
  - SVD는 아이템간 비선형 관계를 충분히 포착하기 못할 수 있음
  - 상관 행렬 계산을 계산 비용이 높아 최적화 필요

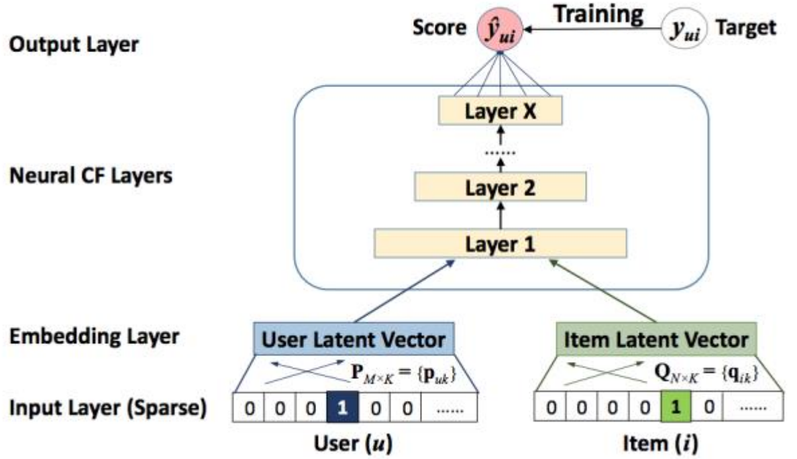
추천 시스템 구현 및 분석

05 Model-based Collaborative Filtering - NCF (Neural Collaborative Filtering)

- 전통적인 협업 필터링 기법은 주로 행렬분해에 기반을 두고 있음

- 2017년에 제안된 Neural Collaborative Filterinv(NCF) 알고리즘은 이러한 접근 방식에 신경망을 결합하여 성능을 높인 Method

- 행렬분해 모델에 비선형 함수를 통합하여 NCF는 사용자-아이템 상호작용간 복잡한 구조를 표현함으로써, 행렬분해 기법의 선형결합 한계를 극복



추천 시스템 구현 및 분석

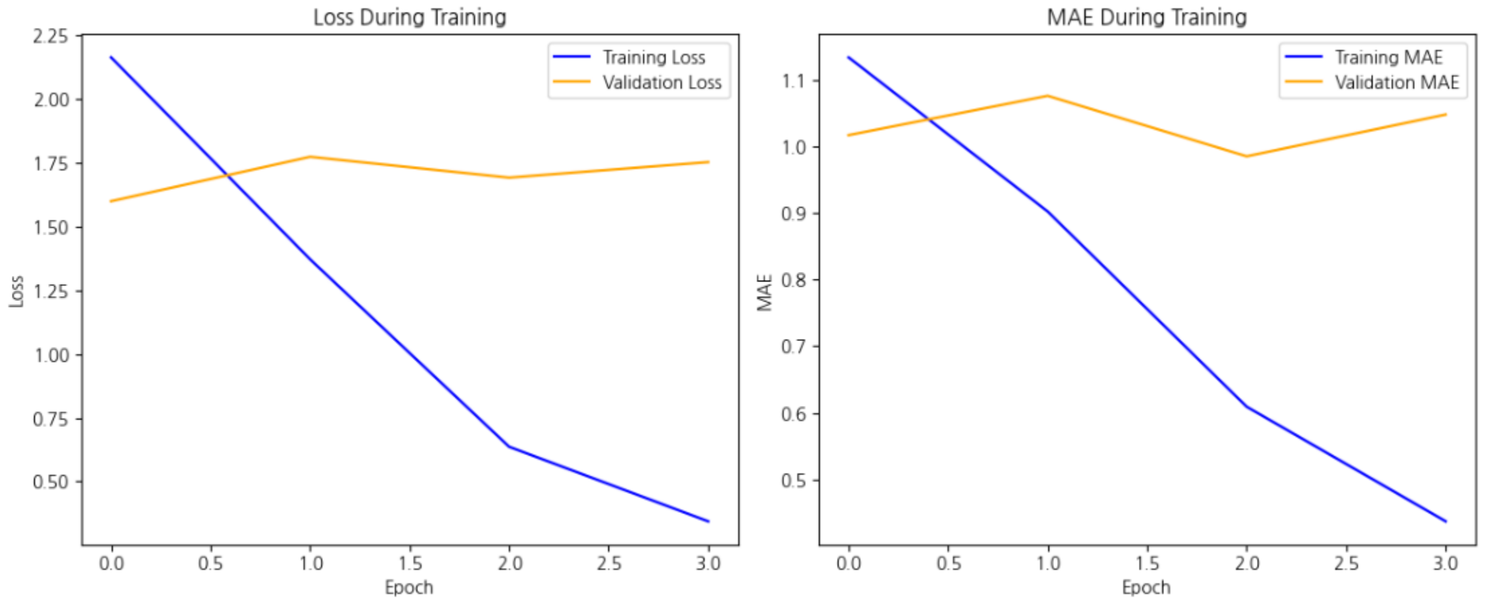
05 Model-based Collaborative Filtering - NCF (Neural Collaborative Filtering)

```python
# NCF 모델 정의
def build_ncf_model(num_users, num_products, embedding_dim=32):
    # Input layers
    user_input = Input(shape=(1,), name='user_input')
    product_input = Input(shape=(1,), name='product_input')

    # Embedding layers
    user_embedding = Embedding(input_dim=num_users, output_dim=embedding_dim, name='user_embedding')(user_input)
    product_embedding = Embedding(input_dim=num_products, output_dim=embedding_dim, name='product_embedding')(product_input)

    # Flatten embeddings
    user_vec = Flatten()(user_embedding)
    product_vec = Flatten()(product_embedding)

    # Concatenate embeddings
    merged_vec = Concatenate()([user_vec, product_vec])

    # Dense layers
    hidden = Dense(128, activation='relu')(merged_vec)
    hidden = Dropout(0.2)(hidden)
    hidden = Dense(64, activation='relu')(hidden)
    hidden = Dropout(0.2)(hidden)
    output = Dense(1, activation='linear', name='output')(hidden)

    # Compile model
    model = Model(inputs=[user_input, product_input], outputs=output)
    model.compile(optimizer='adam', loss='mse', metrics=['mae'])
    return model
```

<NCF 모델 구조>

추천 시스템 구현 및 분석

05 Model-based Collaborative Filtering - NCF (Neural Collaborative Filtering)



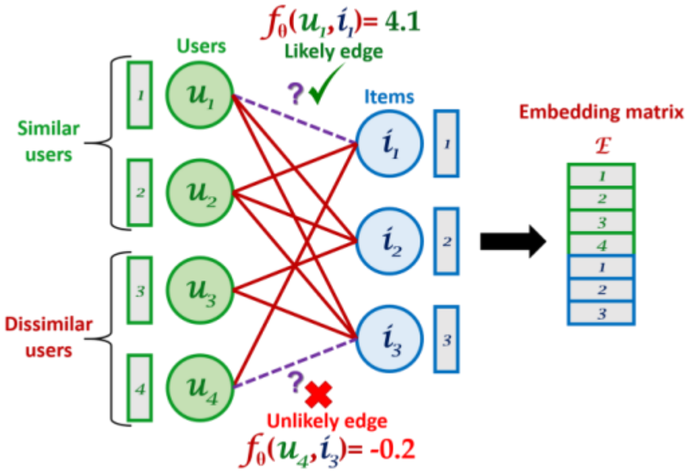
	NCF
epoch	10 (best epoch : 4)
K	K = 10
Precision	0.8
Recall	0.0002
NDCG	0.7105
MSE(test)	1.6116
MAE(test)	1.0204

추천 시스템 구현 및 분석

06 Model-based Collaborative Filtering - LightGCN (Graph Neural Network(GNN) 기반)

- LightGCN은 그래프 기반 추천 시스템에서 사용되는 경량화된 그래프 신경망

- 기존 GCN (Graph Convolutional Network)에서 불필요한 활성화 함수와 중간 레이어를 제거하여 효율적인 추천 시스템 구현



추천 시스템 구현 및 분석

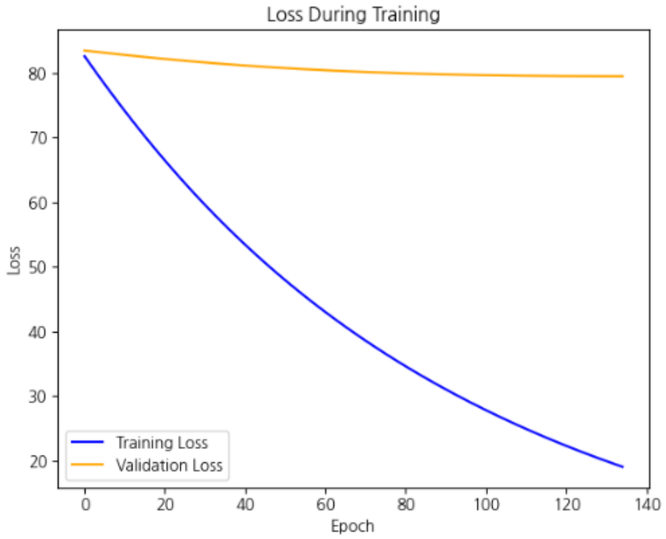
06 Model-based Collaborative Filtering - LightGCN (Graph Neural Network(GNN) 기반)

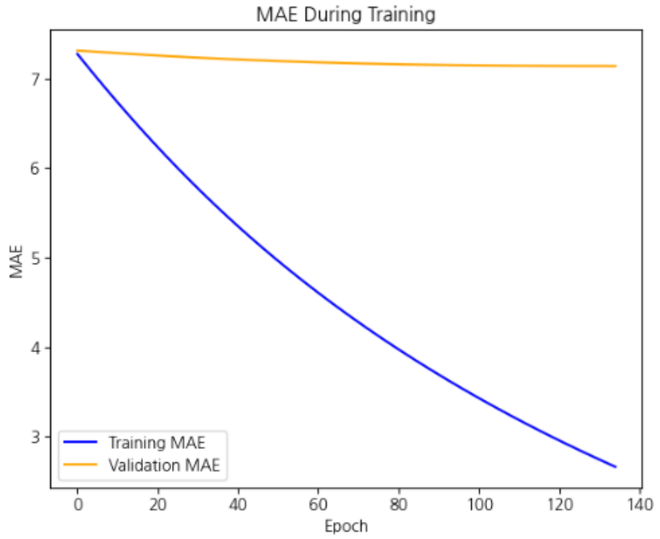
```
# 사용자-아이템 그래프 생성
user_nodes = torch.tensor(ratings['userID'].values, dtype=torch.long)
item_nodes = torch.tensor(ratings['productID'].values + ratings['userID'].nunique(), dtype=torch.long)
edges = torch.stack([user_nodes, item_nodes], dim=0)
```

```
# LightGCN 모델 정의
class LightGCN(nn.Module):
    def __init__(self, num_users, num_items, embedding_dim=64, num_layers=3):
        super(LightGCN, self).__init__()
        self.user_embedding = nn.Embedding(num_users, embedding_dim)
        self.item_embedding = nn.Embedding(num_items, embedding_dim)
        self.convs = nn.ModuleList([GCNConv(embedding_dim, embedding_dim) for _ in range(num_layers)])
        self.num_layers = num_layers

    def forward(self, edge_index):
        x = torch.cat([self.user_embedding.weight, self.item_embedding.weight], dim=0)
        for conv in self.convs:
            x = conv(x, edge_index)
        return x

    def predict(self, user_ids, item_ids):
        user_embeds = self.user_embedding(user_ids)
        item_embeds = self.item_embedding(item_ids)
        return (user_embeds * item_embeds).sum(dim=1)
```

추천 시스템 구현 및 분석

06 Model-based Collaborative Filtering - LightGCN (Graph Neural Network(GNN) 기반)





	LightGCN
epoch	200 (best epoch : 135)
K	K = 10
Precision	0.9
Recall	0.0002
NDCG	0.8611
MAE(Val)	2.6603

추천 시스템 구현 및 분석

07 Model-based Collaborative Filtering 비교 분석

	MF
K	K = 10
Precision	0.0777
Recall	0.7742
NDCG	0.7742
RMSE	4.3219

	NCF
epoch	10 (best epoch : 4)
K	K = 10
Precision	0.8
Recall	0.0002
NDCG	0.7105
MSE(test)	1.6116
MAE(test)	1.0204

	LightGCN
epoch	200 (best epoch : 135)
K	K = 10
Precision	0.9
Recall	0.0002
NDCG	0.8611
MAE(Val)	2.6603

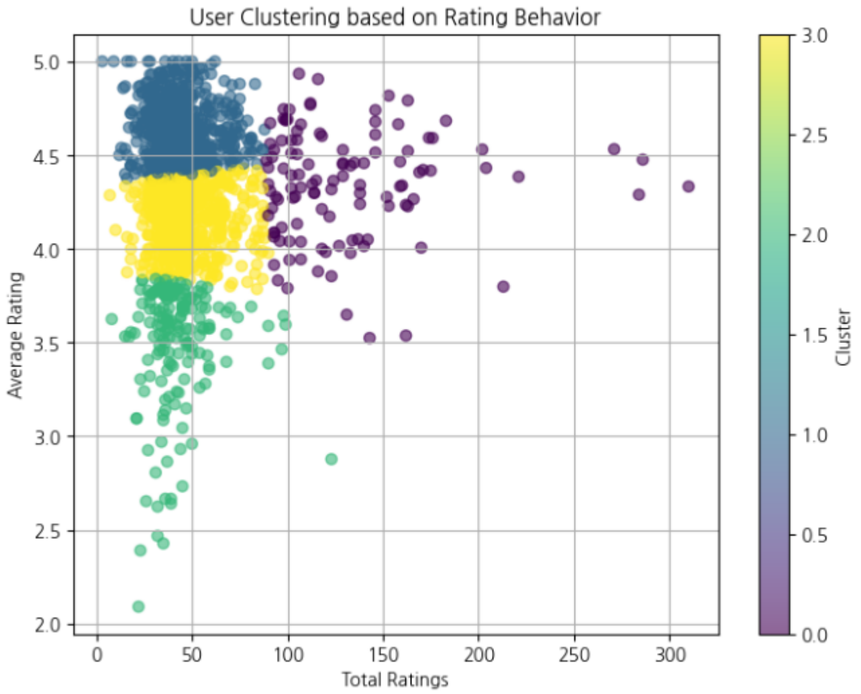
추천 시스템 구현 및 분석

07 Model-based Collaborative Filtering 비교 분석

- **LightGCN**
  - Precision과 NDCG에서 가장 뛰어난 성능을 보임
  - Recall과 MAE는 다소 낮은 성능을 기록
  - 대규모 그래프 데이터를 다루며 추천 품질이 중요한 경우 적합
- **NCF**
  - Precision과 오차(MSE, MAE)가 균형 있게 개선된 모델
  - 빠르게 학습을 완료하고, 중간 수준의 성능을 제공
  - 훈련 시간 대비 적절한 성능이 필요할 때 적합
- **MF**
  - Recall에서 가장 높은 성능을 기록했지만, Precision과 RMSE 성능이 낮음
  - 간단한 행렬 분해 기반 모델로서 구현이 쉬운 반면 성능이 상대적으로 낮음
  - Baseline 모델로 사용하기 적합

결론

01 군집 분석(사용자 그룹 클러스터링) - 활용방안



- **클러스터 0**
핵심 사용자 그룹으로 분류하고, 이들의 피드백을 더 많이 활용해 개선 방안을 도출
- **클러스터 1**
만족도가 높은 사용자들, 신규 사용자에게 추천할 만한 제품이나 서비스를 제안
- **클러스터 2**
불만족도가 높은 사용자들, 이들의 의견을 분석해 개선할 필요가 있음
- **클러스터 3**
평점 수와 만족도가 균형 잡혀 있으므로, 잠재적인 충성 고객 타겟

결론

02 분석 결론

1. 아이템 기반 협업 필터링 (Item-Based Collaborative Filtering)
코사인 유사도 및 피어슨 상관계수를 기반으로 K값을 설정해 성능을 비교
Precision은 상대적으로 낮았지만, Recall이 높아 추천된 항목 중 실제 선호 항목을 많이 포함하는 강점
단점으로는 데이터 양이 많아질수록 계산 복잡도가 증가하는 점이 확인

2. 행렬 분해 기반 협업 필터링 (Matrix Factorization, MF)
행렬 분해를 통해 사용자의 잠재적 선호도를 예측하였지만, Precision과 RMSE에서 상대적으로 낮은 성능
간단한 구조로 빠르게 결과를 도출할 수 있었지만, 복잡한 사용자-아이템 관계를 효과적으로 학습하지 못하는 한계

3. 신경망 기반 협업 필터링 (Neural Collaborative Filtering, NCF)
NCF는 사용자와 아이템의 임베딩을 통해 비선형 관계를 학습
Precision과 MSE에서 균형 잡힌 성능을 보였으며, 빠르게 수렴하였다는 점이 강점
그러나 Recall이 낮게 평가되어, 실제 선호 아이템을 놓치는 경우가 많았음

4. LightGCN (Graph-based Collaborative Filtering)
LightGCN은 그래프 구조를 활용해 사용자-아이템 관계를 학습
Precision과 NDCG에서 가장 뛰어난 성능을 보였으며, 추천 품질이 다른 모델보다 우수

결론

02 분석 결론 - 제언

- 각각의 모델들의 장점을 합한 하이브리드 접근법을 구현하고 싶다.

- 컴퓨팅 성능이 좋지 않아 최소한의 데이터를 가지고 분석하여 추후에는 대규모 데이터셋에 적용해보고 비교해보고 싶다.

- 이 보고서에서 분석한 method 외에도 다른 접근법(샘플링 기법 등) ,
그리고 추천 시스템에서 데이터가 희소해서 나타나는 문제인 콜드 스타트 문제도 함께 풀고 싶다.

참고문헌

Amazon Review Data를 활용한 추천시스템 구현 및 분석

1. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (2020). LightGCN: Simplifying and powering graph convolution network for recommendation. arXiv preprint arXiv:2002.02126. https://doi.org/10.48550/arXiv.2002.02126

2. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering. arXiv preprint arXiv:1708.05031. https://doi.org/10.48550/arXiv.1708.05031

3. Pan, R., Zhou, Y., Cao, B., Liu, N. N., Lukose, R., Scholz, M., & Yang, Q. (2008). One-class collaborative filtering. HP Laboratories Technical Report HPL-2008-48R1.

4. Kaggle. (n.d.). Amazon product reviews: Electronic products user ratings. Retrieved from https://www.kaggle.com/datasets/saurav9786/amazon-product-reviews/data

5. Saurav9786. (n.d.). Recommender system using Amazon reviews. Retrieved from https://www.kaggle.com/code/saurav9786/recommender-system-using-amazon-reviews/notebook#Popularity-Based-Recommendation

Amazon Review Data를 활용한 추천시스템 구현 및 분석

THANK YOU