KNOW 데이터 오류 분석

김지선

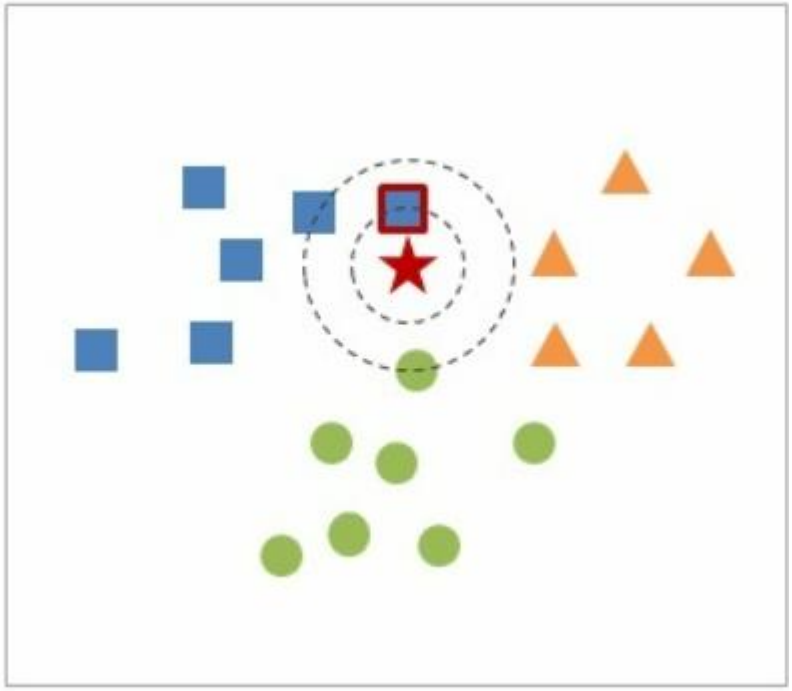
Contents

- KNN이란 무엇인가
- KNN과 CF의 차이
- KNN으로 상위 10개의 오류 분석

KNN 이란?

- 거리기반 분류 분석 모델
- 새로운 데이터가 들어왔을 때 기존 데이터의 어떤 그룹에 속하는지 분류
- 하이퍼 파라미터 K로 범위 조정

  (K=1,3,5 ~)



KNN과 CF의 차이

- 결론적으로 CF가 조금 더 범용성 있는 개념이고 KNN은 CF와 유사도 측정 기준이 다르다.
- CF란 **A와 가장 비슷한 B**를 통해 A에게 추천하는 것
  - 비슷한 기준
  - 사용자: User-based CF
  - 아이템(특성): Item-based CF

KNN 모델이 예측한 정답과 응답 결과

	A	B	C
1	idx	knowcode	
2	0	811201	
3	1	817101	
4	2	31201	
5	3	152101	
6	4	412003	
7	5	415202	
8	6	617101	
9	7	301004	
10	8	110101	
11	9	862101	
12	10	882101	
13	11	132004	
14	12	9999999	
15	13	231401	
16	14	212101	
17	15	29902	
18	16	110101	

- 임의로 20명의 유저의 응답 결과 분석 결과
- 예측한 직업과 응답 항목 중 비교 결과 큰 연관성이 없었음

- 즉 예측을 제대로 하지 못하였다.

KNN 모델에서 오류 분석 결과

- 콜드 스타트
  - 충분한 정보가 수집되지 못해서 새로운 유저들에게 적절 추천 X
- 500/9000로 한 직업에 대한 데이터가 충분하지 않다.
- (이미지)
- 데이터에 적합한 알고리즘이 아닐 수 있다.
- 전처리 과정의 문제
  - 위와 같은 분석하면서 응답 결과에 현재 직업과 이전 직업이 반영되지 않은 점이 성능에 영향을 끼쳤다고 분석

KNN 모델 적용 구조

- 사이킷런은 문자열 값을 입력 값으로 처리하지 않아서 숫자 형으로 변환해야함
- 범주형이 아닌 단순 문자열인 경우 일반적으로 제거

- 문자열 처리 방법
  - 텍스트 칼럼은 임베딩해서 벡터로 유사도를 측정하려 시도
  - 시도 이후 텍스트 칼럼은 제외

1. 원본 데이터에서 텍스트 칼럼과 숫자 칼럼을 분리
2. 모델에 학습하여 Predict

KNN 모델 임베딩 시도 프로세스

1. 텍스트 데이터와 숫자 데이터로 분리
2. 전체 텍스트 데이터에 대하여 토큰화 해서 단어 모델 (Word2Vec) 생성
3. **생성한 모델로 각각의 텍스트 데이터에 대해 모델 돌리기**
4. 임베딩한 텍스트 칼럼과 숫자 칼럼 다시 Merge

KNN 모델 임베딩 시도

1. 전체 텍스트 데이터에 대하여 토큰화 해서 단어 모델 (Word2Vec) 생성

[텍스트 데이터]

	bq4_1a	bq4_1b	bq4_1c	bq5_2	bq19_1	bq30	bq31	bq32	bq33	bq34	bq38_1	
0	자동차도장 기능사			실무교육	생산설비의 자동화로	없다	없다	없다	건설현장 노무직	없다	실업	자동차도장기능사 의 자동화로 없다
1	건축전기설 비기술사				건설 수주가 없어서	없다	매타기, 드라이버, 가위, CAD				건축공학	건축전기설비기술 서 없다 매타기
2	건축전기설 비기술사				신축 건설경기가 좋지 않아서	없다	건축설계표, 오토캐드, 파이프, 보일러				건축공학	건축전기설비기술 좋지 않아서 없다
3	지적기사			측량 및 정보통신기술 교육	업무량 감소, 자동화로 인해서	지적기사	토탈스테이션, SZP	무	지적공무원		환경학과	지적기사 측량 및 업무량 감소, 자동
4	건축전기설 비기술사				부동산 거품 빠지면서 건축 경기도 덩달아 불황	없다	캐드, 엑셀, 건축도면				건축공학	건축전기설비기술 면서 건축 경기
...	...	...	...	...	...	...	...	...	...	...	...	
9481				편집기술에 원리와 이해능력	출판업무 특성상 기획업무는 변함없음	없다	한글, 편집프로그램, 포토샵, 일러스트	없다	마케팅업무	없다	산업디자인	편집기술에 원리와 특성상 기획업무

KNN 모델 임베딩 시도

- 개별 단어에 대해서는 임베딩과 유사도 측정 가능

```
model.wv['CAD']
```

```
array([ 0.66891026,  1.1234144 ,  0.98277366,  0.5410204 ,  0.43546125,
        0.6118243 , -0.01267366, -0.2829244 ,  1.0906429 , -0.32414335,
       -0.3770287 ,  0.18551493,  1.0529205 ,  0.0581194 ,  0.26417744,
       -0.307214  , -1.2799399 ,  0.29205143, -0.52759445, -1.209599  ,
        0.32912517,  0.8966674 ,  1.5432703 ,  0.03188413,  0.1183411 ,
        0.26976827, -0.5301462 , -0.7950405 ,  1.0557704 , -1.2669665 ,
        0.893     ,  0.5178606 , -0.09277198,  0.1719936 ,  0.14988345,
       -0.18121311, -1.0378932 , -1.0847445 ,  0.37125027, -0.37329853,
        0.17710672, -1.6970602 , -0.5829381 , -0.36070868,  0.7317047 ,
       -0.5294191 ,  1.0744421 ,  0.1789279 , -1.4163237 , -0.5518722 ,
        0.9857417 ,  0.5431092 ,  0.19612847, -0.20154056, -1.1067466 ,
       -0.41075835,  0.30748087,  1.7030452 ,  1.1250914 , -1.1394424 ,
       -0.3680961 ,  0.26062432, -0.554122  ,  0.39087662, -0.41557127,
        0.59860474,  0.6059414 , -0.06914414, -0.12047917, -0.45141035,
        0.93001264, -1.0413481 , -1.3162522 ,  0.7742884 , -1.2409852 ,
       -0.26279852, -1.206432  ,  0.9951225 , -0.22200485,  0.07095784,
        0.6597785 ,  0.6935079 ,  1.8022833 ,  0.8730791 , -0.5678337 ,
        0.0675936 , -0.29791185, -1.2945968 , -0.19483049,  0.01119807,
        0.4448029 ,  1.6029941 ,  1.065389  ,  0.81879675,  0.33501637,
       -0.05684851,  0.37486574, -0.01525358,  1.1072345 , -1.4832817 ],
      dtype=float32)
```

```
# 두 단어 유사도 예제1
model.wv.similarity('프로그램','CAD')
```

```
0.8120801
```

```
# 두 단어 유사도 예제2
model.wv.similarity('프로그램','chef')
```

```
-0.024864515
```

```
# 두 단어 유사도 예제3
model.wv.similarity('프로그램','컴퓨터')
```

```
0.8990118
```

KNN 모델 임베딩 시도

- **생성한 모델로 각각의 텍스트 데이터에 대해 단어 모델 돌리기**

즉 개별로 단어를 토큰화하고 벡터화하는 것은 성공했으나

- 단어사전에 없는 Unknown 데이터 처리 문제
- 토큰화해서 나오는 토큰 개수가 달라서 shape 에러 발생

```
  0%|          | 0/9486 [00:00<?, ?it/s]bq4_1a      [자동차도장기능사]
bq4_1b              []
bq4_1c              []
bq5_2           [실무교육]
bq19_1    [생산설비의, 자동화로]
bq30              [없다]
bq31              [없다]
bq32              [없다]
bq33       [건설현장, 노무직]
bq34              [없다]
bq38_1            [실업]
Name: 0, dtype: object

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(s
```

텍스트 칼럼 임베딩 시도 결과

- 텍스트 칼럼에 주요한 정보가 있었는데 그 과정에서 막혀서 이를 제외하고 학습한게 낮은 정확도에 크게 작용했을거라 판단됨

Sol1) 다른 임베딩 방법 적용

- 단어 칼럼을 모두 띄어쓰기 단위로 합치기 -> 문장 칼럼
- 문장 칼럼을 ㄱㄴㄷ순으로 정렬하고
- 통째로 그 문장칼럼끼리 유사도를 구한 new feature 생성

Sol2) 문자 칼럼 인코딩

- 문자열 칼럼을 토큰화한 상태(단어,조사)에서 각각 원핫 인코딩

KNN 성능 개선 방법

- 랜덤 포레스트 적용 -> Memory Error
- 텍스트 칼럼을 전처리 재시도
- 이진 분류 문제로 변경 후 유사도 측정

계속 하는 이유

- 성능 안 나온다고 그대로 방치하는게 아닌 여러가지 방법을 시도해 보고 아래와 같은 점을 얻음
  - 사이킷런의 전처리 과정을 정확히 파악 (문자열/숫자)
  - 전처리 과정의 중요성을 깨달음
  - 단어 모델 사용 가능(자연어 처리)
  - Pandas 활용 능력 UP
  - 사이킷런 모델 활용 능력 UP

다음 주제

- 무비렌즈 데이터 MF로 분석
- GPT, BERT 등 유명한 언어모델 구현해보기