문제 1.CF (Collaborative Filtering) 방법론의 단점인 cold start problem과 popularity bias problem에 대해 설명하고 CB (Contents Based) 방법론이 이를 어떻게 해결할 수 있는지 설명해주세요. (선택 질문) 두 문제가 정말로 전체 성능에 영향을 주는 경우는 어느때일까요? 유저/아이템 전체 숫자 및 신규 유저/아이템 숫자, 유저/아이템의 이탈률 혹은 수명을 바탕으로 설명해주세요.

* Cold Start problem - CF는 유저 데이터를 사용해 추천을 하기 때문에 새로운 유저나 아이템을 추천할 수 없다는 문제

* Popularity bias - 모든 사람들이 좋아하기 때문에 실제 내용이 전혀 관련이 없더라도 추천이 되는 경우를 의미합니다. 예를 들어 음악 추천에서 항상 아이돌 음악이 나오는 경우가 있습니다.
> 만약 유저 숫자가 충분히 많고, 유저 및 아이템이 거의 새로 추가되지 않는다면 Cold start problem이 크게 중요하지 않을 수도 있습니다. 그러나 만약 아이템의 수명이 매우 짧고 유저의 리텐션이 낮은 경우에는 cold start problem이 심각해지며, 덩달아 popularity bias도 심해집니다. 이는 서비스에 나쁜 영향을 줄수도 큰 영향을 주지 않을 수도 있습니다. 개인의 취향이 아주 확실한 서비스에서는 (예: 동영상, 음악) popularity bias가 사용자들에게 나쁜 경험을 줄 수도 있지만, 개인의 취향이 덜 확실한 서비스에서는 (예: 뉴스) popularity bias가 사용자들에게 나쁜 경험을 줄 확률이 낮을 수 있습니다.

* CF: 유저 데이터를 사용해 추천해서 좀 더 유저들의 취향을 잘 반영할 수 있다, 직접 컨텐츠를 몰라도 유저들이 선호하는 방식을 통해 컨텐츠의 내용을 몰라도 비슷한 컨텐츠가 추천되는 경우가 많다. 그러나 cold start와 popularity bias라는 이슈가 있다.

* CB: 유저 데이터를 사용하지 않고, 컨텐츠만 사용한다. 그래서 cold start와 popularity bias 이슈가 없지만, 유저 선호를 반영할 수 없어서 성능은 조금 떨어질 수 있다

문제 2.Text 데이터 (문장 데이터)의 유사도를 측정하기 위해서는 먼저 주어진 문장을 vector로 만들어야 합니다. 텍스트를 벡터로 만들기 위해서는 어떻게 해야 할까요? 방법론은 임의의 방법론을 선택했다고 가정합니다. (예: CBoW, RNN등) (Hint: 다음 단어들을 사용해 설명해주세요 token, vocabulary, embedding) (선택 질문) 만약 token을 단어로 고를 경우 어떤 문제가 있을 수 있을까요? 이를 어떻게 해결할 수 있을까요?

(1) 먼저 주어진 문장을 token 단위로 쪼갭니다. Token은 정의하기 나름인데, 공백 단위로 자를 수도 있고 단어 단위로 쪼갤 수도 있고 음절단위로 쪼개거나 sub-word단위 심지어는 character 단위까지 자르는 것도 가능합니다.

(2) Vocabulary에서 token에 해당하는 index를 꺼내옵니다 - 이를 encoding이라고 합니다.

(3) Token하나를 vector로 바꾸어줍니다. 이를 embedding이라고 하는데, 가장 간단한 예로는 one-hot-embedding이 있습니다. 이제 이 데이터를 사용하여 CBoW, RNN 등의 방법론을 학습하여 문장을 벡터로 만들어줍니다.
> 단어를 token으로 잡을 경우, 새로운 단어가 추가될 때 마다 학습을 다시 해야한다는 단점이 있습니다. 또한 오타가 난 경우에도 다른 token으로 인식한다는 단점이 있습니다. 이를 막기 위해서는 sub-word나 character 단위로 embedding을 만드는 방법론이 있긴 하지만 아직 연구 단계입니다.

****어떤 데이터이든 (텍스트이든 이미지이든 음성이나 음악 어떤 것이든) 유사도를 측정하기 위해서는 데이터를 벡터로 만드는 과정이 꼭 필요하다.

문제 3.조경현 교수님의 렉쳐에서 소개된 CBoW와 RNN 기반 방법론을 사용해 text similarity를 측정하였을 때 각각의 방법론의 장단점은 무엇일지 설명해주세요. (Hint: 텍스트의 순서, 연산 속도)

* CBoW: 장점 - 빠르고 대략 잘 동작한다
단점 - Token의 순서를 고려하지 않으므로 깊은 이해가 어렵다.

* RNN: 장점 - Token의 순서를 잘 고려할 수 있고 모델이 복잡하여 표현할 수 있는 모델 파워가 강하다.
단점 - 새로운 text가 들어오면, 이를 벡터화하기 위해 RNN forward를 진행해야 하는데 이것이 매우 느리다. 빠르고 신속한 추천이 필요한 경우에는 큰 단점이 될 수 있다.
RNN을 쓰는 경우에는 새로운 텍스트가 들어오면 매 번 RNN 연산을 다 해줘야지만 벡터 값을 얻을 수 있어서 다소 비효율적, 바로바로 연산이 필요한 경우에는 적절하지 않다

문제 4.(Open Question) 다음 세 가지 경우에 대하여 어떤 방식으로 추천 시스템을 구축하는 것이 효과적일까요? CF와 CB, Hybrid 등의 그 동안 배운 방법론들을 기반으로 설명하여봅시다.
(1) 유저 데이터가 매우 적은 콘텐츠 추천 스타트업 (유저 숫자는 수백명, 콘텐츠는 수천개라고 가정합니다)
(2) 유저 숫자가 매우 많고 콘텐츠의 변화량이 적은 대형 서비스 (유저 숫자는 수백만명, 콘텐츠는 수천개라고 가정합니다)
(3) 유저 숫자는 매우 많으나 콘텐츠가 매일 매일 새로 등장하는 뉴스 추천 (유저 숫자는 수백만명, 콘텐츠는 매일매일 수백개씩 생겨난다고 가정합니다)*

1 – CB : 유저 데이터가 매우 적기 때문에 Matrix 자체가 매우 과도하게 sparse하다, MF 계열은 matrix가 sparse하면 할 수록 쓰기 어렵다.

2 – CF : Matrix가 덜 sparse한 상황, 또 CF의 가장 큰 문제점 중 하나인 cold start problem이 발생하지 않는 상황
3 – Hybrid : CF만 쓰기에는 cold start 이슈가 매우 빈번하게 발생하는 셋팅. 반면 Matrix 자체는 너무 sparse하지 않으니 cold start 부분만 잘 커버가 된다면 CF가 역할을 잘 해 줄 거라고 기대.