Session- Based Recommendation with RNN

Abstract

우리는 RNN을 새로운 도메인에 적용했다, 추천 시스템에.

현실 추천 시스템은 종종 문제에 직면한다 짧은 세션 기반의 데이터에 기본적인 추천을 하는 긴 유저 히스토리 대신에

이러한 상황에서 빈번하게 찬양되는 MF 접근법은 정확하지 않다

이 문제는 보통 실전에서는 극복 되지만 item-to-item 추천에 의지해서, 유사한 아이템 추천 으로

우리는 주장한다. 전체 세션을 모델링 함으로 써, 더 정확한 추천이 가능할 수 있다

우리는 그래서 세션 기반 추천을 위한 RNN 기반의 접근법을 제시한다

우리의 접근법은 물론 고려한다 실용적인 측면을 업무에 대해 그리고 소개한다 몇 가지의 변경할 부분을 전통적인 RNN에 대해 와 같이 loss 함수의 ranking을 하는 것 특정 문제에 좀 더 잘 맞도록

실험 결과는 두가지 데이터 셋에 대한 결과로 개선된 점을 보인다 널리 쓰이는 접근법 들에 대해

Introduction

세션 기반의 추천은 상대적으로 진가를 인정 받지 못한 문제다 머신 러닝과 추천 시스템 커뮤니티에서

많은 이-커머스 추천 시스템 (특정 작은 소매업에서) 그리고 거의 모든 뉴스와 미디어 사이트들은 일반적으로 추적하지 않는다 특정 유저의 아이디를 유저가 방문했던 그들의 사이트에서 긴 기간 동안

하지만 cookies이나 browser finger print가 공급 할 수 있다. 어느 수준의 유저에 대한 인식을 (recognizability) , 그러한 기술들은 종종 충분히 믿을 만하지 않고 프라이버시 문제를 끌어 올린다

만약 추적이 가능 하더라도 , 많은 유저 들은 오직 하나나 두개의 세션을 가진다. 작은 이 커머스 사이트, 특정 도메인에서는 유저의 행동이 종종 세션 기반의 특성을 보이는

그래서 세션의 서브 시퀀스 같은 유저 의는 독립적으로 다뤄져야 한다.

결과적으로 , 거의 모든 세션 기반 추천 시스템 들 이-커머스를 위해 배포되는 상대적으로 간단한 메소드를 기반으로 한 유저 프로파일을 사용하지 않는다 예를 들어 item-to-item 유사도, co-occurrence 그리고 transition probabilities를 사용한다고

효과적이지만, 이러한 방법들은 오직 마지막 클릭이나 유저의 선택을 취한다 과거 클릭 정보를 무시하고

가장 흔한 메소드 추천 시스템 들에서는 factor 모델이나 neighborhood 모델 들이다.

Factor 모델들은 작동 되게 된다 분해에 의한 sparse 한 유저-아이템 상호작용 매트릭스 정해서 d차원 벡터를 각각의 아이템과 유저 데이터 셋 내 에서

추천 문제는 따라서 다뤄진다 매트릭스 completion / reconstruction 문제로 latent factor 벡터 들은 그러면 사용된다 채우기 위해 비어있는 entries를, 예를 들어 dot-product를 한다 할당되는 유저-아이템 latent factors 에

팩터 모델들은 적용하기 힘들다 세션-기반의 추천을 그 이유는 유저 프로 파일이 없기 때문에

반면에, neighborhood 방법은 의존한다 계산하는 것에 유사도를 아이템 간 (유저나) 기반 해서 아이템 co-occurrences(유저 프로파 일이나) 세션 내

Neighborhood 메소드 들은 사용될 수 있다. 광범위 하게 세션 기반의 추천에서

과거 몇 년 간 보였다 엄청난 성공을 딥 러닝이 많은 분야에서 이미지나 음식 인식과 같은 구조화 되지 않은 데이터는 처리 될 수 있다 통해서 몇 개의 Conv와 표준 레이어 (rectified unit)에 의해

Sequential한 데이터 모델링은 최근에 물론 많은 관심에 이끌려 다양한 특성의 RNN들은 이러한 데이터 타입을 위한 모델 선택 되었다

어플리케이션 들은 시퀀스 모델링의 범위가 test-translation 부터 image captioning의 conversation 모델링 다

RNN들이 적용되어 지는 동안 앞서 언급된 도메인에 두드러지는 성공과 적은 관심을 받으면서 추천 시스템 역에 소모 되어져 왔다.

이러한 와중에 우리는 주장한다 RNN들은 적용될 수 있다고 세션-기반의 추천에서 두드러지는 결과를 보이면서

우리는 이슈 들을 다루며 뭐 모델링 할 때의 sparse sequential data 이슈와 또한 적용을 RNN 모델을 추천 시스템 세팅에 도입해서 새로운 ranking loss 함수를 이러한 모델들을 학습하기 위한

세션 기반의 모델 문제는 몇가지 공통점을 가진다 자연어 처리와 모델링을 할때 시퀀스 길이와 어느 정도 타협을 본다는 점에서

세션 기반 추천에서 우리는 고려가 가능하다 유저가 클릭한 첫번째 아이템 언제 방문했을 때 웹 사이트를 첫번째 입력으로 RNN의 그리고 하고 싶을 것이다 쿼리를 모델을 기반 해서 첫 번째 입력을 추천 시스템을 위한

각 연속적인 유저 클릭은 생성할 것이다 추천 결과를 모든 이전 클릭에 기반 해서

일반적으로 아이템 셋 추천 시스템으로 부터 선택된 은 수천 개 중 몇 십 개나 추천 개 중 몇 백 개가 될 수 있다

떨어져서 아이템 셋의 거대한 사이즈에서, 또 다른 챌린지는 클릭-스트림 데이터 셋은 일반적으로 꽤 크다 그래서 학습 시간과 확장성이 매우 중요하다

대부분의 정보 검색이나 추천 세팅 들에서 우리는 관심이 있다 초점을 맞추는 것 모델링 파워에 상위에 위치한 아이템들 예를 들어 유저가 흥미를 가질 만한, 마침내 우리는 RNN을 학습하기 위해 ranking loss function을 사용한다

Related WORK

대부분의 작업 추천 시스템의 영역 내부에서는 초점을 맞춰 왔다 모델에 유저 식별이 가능하고 깔끔한 유저 프로 파일이 구성 될 때 잘 동작하는

이러한 세팅 에서 Matrix Factorization 메소드와 Neighborhood 모델들은 학문적으로는 지배적이고 온라인 에서 채택이 되었을 수 있다.

주요한 접근법은 채택 되는 것이다. 세션 기반의 추천과 자연적 결과 문제에 대한 비어있는 유저 프로 파일은 item-to-item 추천 접근법 이러한 환경에서 아이템 에서 아이템 유사 매트릭스는 미리 계산된다 사용 가능한 세션 데이터에서 부터 아이템은 세션 내에서 흔하게 같이 클릭 되고 유사한 아이템으로 간주 된다.

이 유사도 매트릭스는 후에 간편하게 사용된다 세션 동안 추천하기 위해 가장 비슷한 아이템들을 유저가 현재 클릭 했었던 아이템을

간단하지만, 이 메소드는 효과가 증명 되었고 많이 채택 되었다.

효과적이지만, 이러한 메소드 들은 유저의 마지막 클릭만 사용하고 과거 클릭 들의 정보는 무시한다

조금 다른 접근법 세션 기반의 추천 은 MDPs(Markov Decision Processes) 다

MDPs는 모델인데 sequential stochastic descision 문제이다

MDPs 들은 정의된다 4개의 튜플(S, A, Rwd, tr >로 정의 되는데 S는 상태 셋, A는 액션 셋, Rwd는 보상 함수 그리고 tr은 상태 전이 함수 다.

추천 시스템에서 액션은 추천과 같아 질 수 있고 가장 간단한 MPDs 는 필수적으로 1차 마르코프 체인으로 다음 추천이 간단하게 정해질 수 있는 기반 해서 transition 확률로 아이템 간의

주요 이슈 적용 마르코프 체인을 세션 기반 추천에서는 상태 공간이 빠르게 관리할 수 없게 된다는 점이다 포함하려고 할 때 모든 가능한 유저의 선택 시퀀스를

확장된 버전의 일반적인 Factorization 프레임워크 (GFF)은 추천을 위한 세션 데이터가 사용 가능하다

이것은 세션을 모델링한다 이벤트들의 합으로

두 종류의 latent representation을 아이템을 위해 사용하는데 하나는 아이템 그 자체를 표현하고, 다른 하나는 아이템들을 세션의 일부로 표현한다

세션은 그러면 표현된다 세션의 한 부분으로 표현된 아이템 피처 벡터 들의 평균치로

그러나 이 접근법은 고려하지 않았다 세션 내에서의 순서를

추천 에서의 딥러닝

첫 번째로 관련된 메소드 뉴럴 네트워크 문서 상으로는 RBM이다 CF를 위한

이 연구에서 RBM은 사용되었다 모델링 하는데 user-item 상호작용을 그리고 추천을 했다

이 모델은 CF 모델에서 가장 잘 도착하는 모델 중 하나로 비추어 지고 있다

딥 모델은 추출하기 위해 사용 되었다 구조화 되지 않은 컨텐츠인 음악이나 이미지 같은 데서 특징을 추출할 때 체계적으로 하기 위해서

CNN이 사용된 적도 있다 피처 추출을 위해서 음악 파일의 그리곤 factor 에 사용 되었다

좀 최근에는 도입했다 좀더 일반적인 접근법을 딥러닝을 추출하는데 사용하는 것 일반적인 컨텐츠 피처로 어떤 타임의 아이템으로 부터 이러한 피처 들은 표준화 되어있는 CF 모델과 결합되어 추천 성능을 개선했다

이런 접근법은 보여 질수 있다 특별히 유용 하다고 유저-아이템 상호작용 정보가 충분하지 않을 때에 (충분한 적이 없다.)

3 RNNs로 추천하기

RNN은 시퀀스 데이터를 위해 고안된 모델이다.

주요한 차이점 RNN 과 일반적인 피드 포워드 딥 모델 간 은 존재 여부 내부 hidden state 유닛 내 네트워크를 구성하는

표준 RNN들은 업데이트함 그들의 hidden state h를 다음과 같은 업데이트 함수로

$$h_t = g(Wx_t + Uh_{t-1})$$

g는 평평하게 하거나 튀게 하는 함수 다. logistic sigmoid 함수와 같은 x_t는 시간 t에서의 입력 유닛이다.

RNN 의 출력은 확률 분포다 시퀀스의 다음 엘리먼트의 주어진 현재 상태 h_t에 대해서

GRU는 좀 더 정교한 모델이다 RNN 유닛의 목표로 한다 vanishing gradient 문제를 해결하기 위한

GRU 게이트는 필연적으로 학습한다 언제 그리고 어떻게 얼마나 hidden state 유닛을 업데이트 할지를

GRU의 활성화는 선형 보간법이다 간의 이전 활성화와 후보 활성화 위에 $\hat{h}_t$

GRU 모델 커스터마이징

우리는 사용했다. GRU 기반의 RNN을 우리의 모델에 세션 기반의 추천을 위해.

네트워크 입력은 실제 상태다 세션의 그렇지만 출력은 아이템이다 다음 이벤트의 세션 내의

세션의 상태는 둘다 가능하다 실제 이벤트의 아이템 이거나 결국 세션 내에서의 이벤트로

전자 경우에는 N 분의 1 인코딩이 사용 되었었다

입력 벡터의 길이는 아이템 들의 길이와 같고 오직 배치 된다면 활성화된 아이템 하나만 1이 되고 나머지는 0이 되도록

후자의 세팅은 사용한다 이러한 표현 식의 가중치의 합으로 이벤트 들은 수치가 감소 할 것이다. 더 빨리 일어난 이벤트들은

안정성을 위해, 입력 벡터들은 Normalized

우리는 예상한다 이렇게 하는 것이 도움될 거라고 그 이유는 메모리 효과를 강화 하기 때문이다 굉장히 local ordering 한 제약을 강화하는 것 잘 잡히지 않는 long memory의 RNN에서는

우리가 실험을 해봤지만 추가적인 임베딩 레이어를 추가하면서, 결국은 N 분의 1 인코딩이 항상 잘 동작함

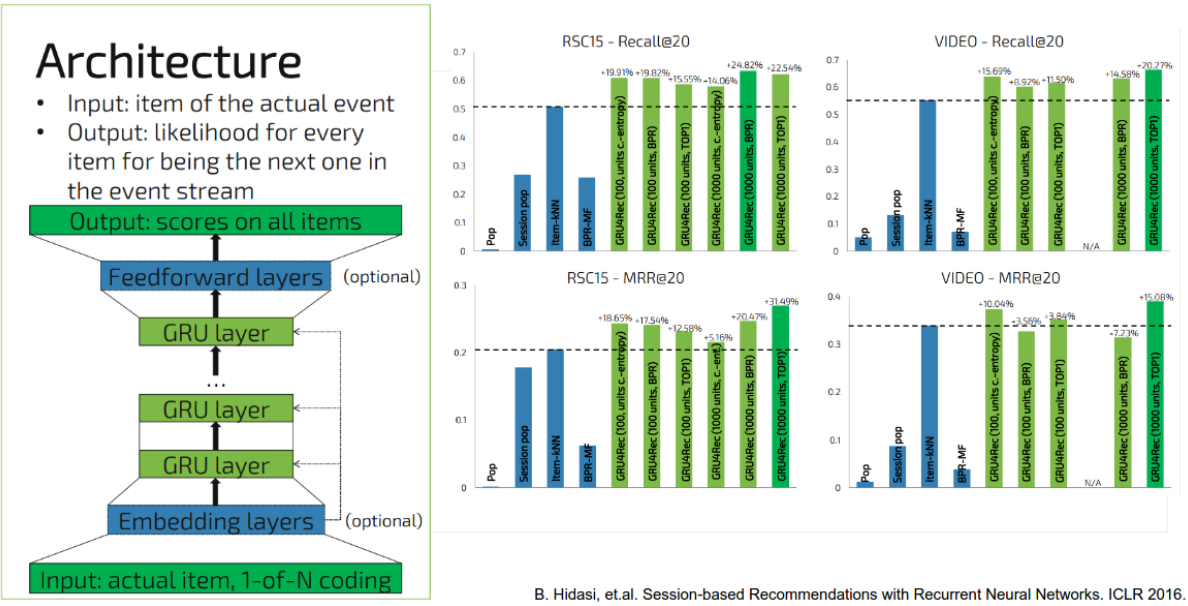
네트워크의 코어는 GRU 레이어고 추가적인 FeedForward 레이어는 추가 될 수 있지만, 마지막 레이어와 출력 사이에서

출력은 예측한다 아이템들의 선호도를

다음이 될 likelihood는 세션 내에서 각 아이템들에 대해

다중 GRU 레이어 들이 사용 될 때에는 hidden 상태인 이전 레이어는 다음 것의 입력이 된다

입력은 선택적으로 연결 될 수 있는데 GRU로 네트워크가 깊어지도록, 이렇게 하는게 성능을 향상 시킨다



B. Hidasi, et.al. Session-based Recommendations with Recurrent Neural Networks. ICLR 2016.

추천 시스템이 RNN 분야에서는 중요한 application 영역이 아니기 때문에, 우리는 변경을 했다 기본 적인 네트워크를 task에 더 잘 맞도록

우리는 물론 고려했다 실용적인 측면을 그래서 우리의 솔루션이 live 한 환경에서도 적용될 수 있었다

SESSION-PARALLEL MINI-BATCHES

RNN들 자연어 처리 Task들은 보통 사용한다 in-sequence mini-batches를

예를 들어 흔하게 사용한다 슬라이딩 윈도우 방식을 문장들의 단어들에 대해 그리곤 이러한 윈도우 화한 조각들을 둔다 다음의 각각 다른 것들에게 미니 배들을 형성하기 위해

이것들이 우리의 task에는 맞지 않는다, 그 이유는

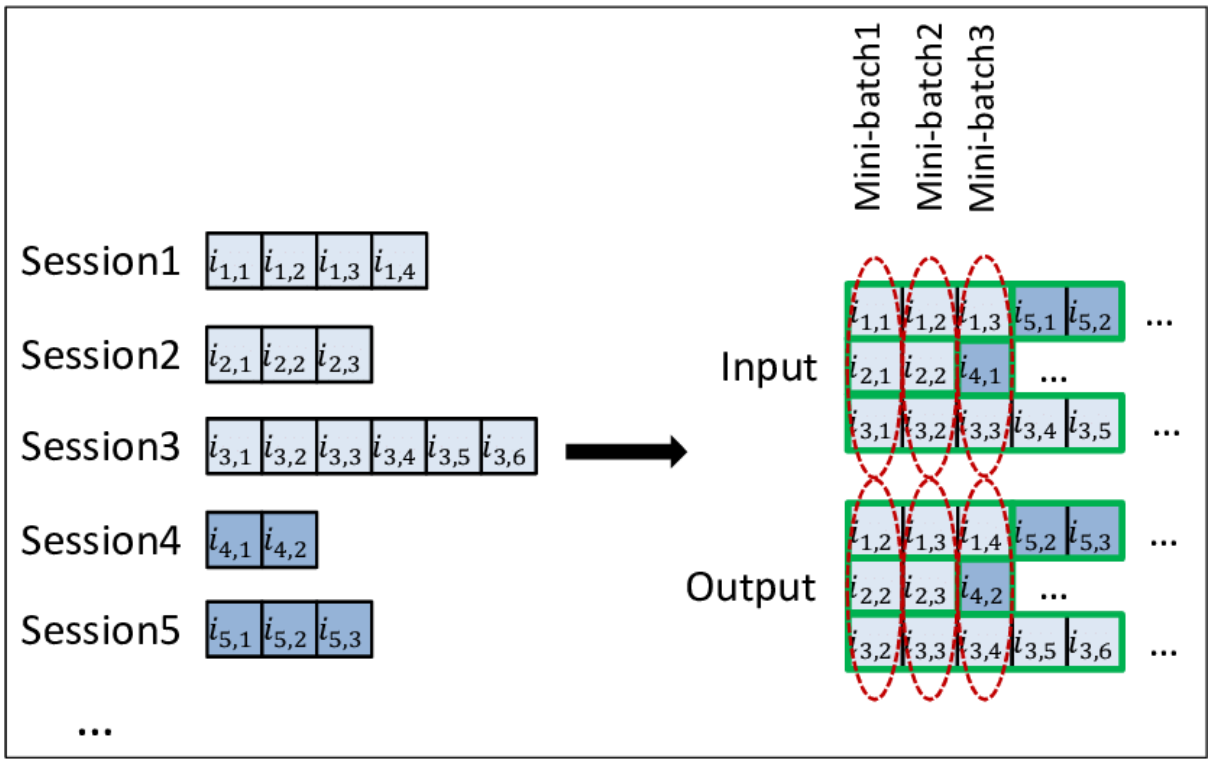
(1) 세션의 길이 들이 너무 다르다 : 수백 개 짜리도 있고 단 두 개 짜리도 있다

(2) 우리의 목적은 알아 내는 것이다. 어떻게 세션이 변화할 것인지 시간에 따라서 그래서 조각들로 자를

그러므로 우리는 병렬 세션의 미니 배치를 사용한다

만약 어떤 세션 이나 끝난다면, 다음 가능한 세션이 대체한다

세션 들이 독립적이라고 가정된다 , 그래서 우리가 스위치 가 발생을 할 때 적절한 hidden state로 리셋 한다



SAMPLING ON THE OUTPUT

추천 시스템 들은 전체 아이템 들의 숫자가 커지면 유용하다

중간 정도 크기의 인터넷 상점에서도 수만의 범위를 가지지만, 사이트 들이 커져 갈 수록 줄어들지 않는다

계산하는 것 점수를 각각의 아이템을 각각의 스텝에서 는 만들수 있다 알고리즘의 스케일 조정을 아이템의 수와 사건의 수의 곱으로

실제 환경에서는 사용을 못할 수 도 있다

그래서 우리는 출력을 골라야 (sample) 하고 오직 아이템들의 작은 부분 집합을 위해 점수를 계산해야한다

이것이 물론 일부의 가중치만 업데이트 되게 만들 수 도 있지만. 원하는 출력 외에도, 우리는 필요하다 점수 계산하는 것을 몇개의 negative example들을 그리고 변경하는 것이 가중치를 원하는 결과가 높은 랭크를 가지도록

자연적인 보간법 임의로 누락된 사건에 의한 은 유저가 아이템의 존재를 몰라서 상호작용이 안 일어나는 사건이다 그러나 낮은 가능성으로 유저가 아이템을 알고 있지만 선택을 안해서 발생할 수 도 있다. 싫어하니까

아이템의 인기가 있는 아이템 일 수록, 더 많은 확률로 고객이 알 수 있고, missing event가 되면 고객이 싫어한다는 뜻이다

그러므로 우리는 아이템들을 샘플링 해야 한다 상품들의 인기도에 비례해서

대신에 생성하기 분리된 샘플 들을 각각의 학습 예시로 부터 우리는 상품을 다른 배치에서 사용한다. negative example 으로서

이 방법의 장점은 줄일 수 있다는 점이다 계산 시간을 샘플링을 건너뜀으로

추가적으로 이점이 있다 구현 측면에서 코드를 만들 때 덜 복잡하게 만들어 매트리스 연산을 가속화 한다

게다가 이러한 접근법은 물론 인기-기반의 샘플링이다 그 이유는 likelihood of an item은 다른 학습 예제인 미니 배치에서는 인기도의 한 부분이기 때문이다

RANKING LOSS

추천 시스템의 코어는 신뢰 기반 아이템 랭킹이다

비록 task가 해석될 수 있다 분류 task로, Learning-To-Rank 접근법은 일반적으로 접근법 들보다 탁월한 성능을 낸다

랭킹은 point-wise, pair-wise 혹은 list-wise가 될 수 있다.

Point-Wise 랭킹은 예측한다 점수나 랭크를 아이템에 대해 각각 독립적이고 loss도 정의되어 관련 있는 아이템의 랭크가 낮아야 한다

Pair-Wise 랭킹은 비교한다 점수 혹은 랭크를 Positive 와 Negative 상품의 쌍의 그리고 loss는 강제한다

Positive한 상품의 랭크는 낮아야 한다고 negative한 상품 보다는

Listwise 랭킹은 사용 한다 스코어와 랭크를 모든 아이템에 대한 그리고 그것들을 비교한다 완벽한 결과와

이것이 정렬을 포함 함으로써, 계산적으로 더 비싸지고 많이 사용 되지 않는다

물론 관련성 있는 아이템만 있다면 - listwise 랭킹은 pairwise ranking으로 해결 될 수 있다

우리는 포함 했었다. pointwise와 pairwise 랭킹 losses 들을 우리의 솔루션으로

우리는 발견했다 pointwise 랭킹은 이 네트워크에서는 불안정하다.

반면에 pairwise 랭킹 loss는 잘 동작한다

BPR: 베이지안 개인화 추천 (2009)은 MF 메소드이다. pairwise ranking losses를 사용하는

비교한다 점수를 positive와 sampled negative 아이템들의

여기서 우리는 비교한다 점수를 positive 아이템과 함께 얼마의 sampled item들에 의한 그리고 사용한다 그것들의 평균을 loss로 사용한다

한 세션 내에서 주어진 포인트 에서의 loss는 정의된다

$-$모든$NegativeSample$에대해$(PositiveSampleScore - NegativeSampleScore)$의평균

을 최소화 하도록 여기서 Positive Sample Score를 위한 상품은 시퀀스 내 의 바로 다음 상품이다

TOP1: 이 랭킹 로스는 우리가 도출 했다 이 Task를 위해

이 방법은 관련 아이템의 상대적인 랭크를 정규화 해서 근사 한다

연관된 아이템의 상대적인 랭크는 주어졌다

모든 Negative Sample에 대해 NegativeSampleScore와 PositiveSampleScore를 비교한 결과들의 평균

시그모이드를 사용해서 비교 결과를 가정한다

최적화 이것을 위해 변경 해야 한다 파라미터 들을 그러면 i를 위한 점수는 높을 수도 있다

그러나 이것은 불안정하다 특정 positive 상품들도 negative 예제로 활동할 수 있고 그에 따라 점수가 엄청나게 높아질 수 가 있다

이런 걸 피하기 위해 negative 예제들의 점수를 0에 가깝게 강제로 매기려고 한다

이것은 negative 아이템들의 상품에 대한 당연한 기대다

그래서 우리는 regularization 이라는 용어를 loss에 추가 했었다

중요하다 이러한 용어는 같은 범위로 상대적인 랭크로 그리고 비슷하게 동작하는 것이

마지막 loss 함수는 다음과 같다

act(모든$NegativeSample$에대해($Negative\ Sample\ Score - Positive\ Sample\ Score$)의평균) + act($NegativeSampleScore$의제곱)

4. 실험

제시된 RNN에 대해 유명한 베이스 라인 으로 평가했다 두 데이터 셋으로