17/11/2

협업필터링(Collaboratice Filtering)

많은 [사용자](https://ko.wikipedia.org/wiki/%EC%82%AC%EC%9A%A9%EC%9E%90_(%EC%BB%B4%ED%93%A8%ED%8C%85))들로부터 얻은 기호정보(taste information)에 따라 사용자들의 관심사들을 자동적으로 예측하게 해주는 방법

최근은 CBF(Content-based Filtering)와 CF를 결합한 하이브리드 방식을 사용한다.

추천의 기본 원리는 과거는 곧 미래다. 추천을 위해서 필요한 데이터는 유저들의 행위기록이라고 할 수 있다.

필요한 데이터: 유저레이팅, 영화정보(메타데이터), 사용자의 특성(사용자 데모그래픽), 소셜관계 데이터

모델기반 CF : 나이브 베이지언이나 뉴럴네트워크

메모리기반 CF : 유저-아이템 레이팅을 모두 메모리에 올려두고 유저-유저 또는 아이템-아이템의 관계 (유사도)를 계산

훈련데이터는 연대순으로 정렬 되어있다.그래서 유효성 검사할 때 아래 2.5min 행을 지워라

def isrc\_to\_year(isrc):

if type(isrc) == str:

if int(isrc[5:7]) > 17:

return 1900 + int(isrc[5:7])

else:

return 2000 + int(isrc[5:7])

else:

return np.nan

I recommend the 3rd course of deeplearning.ai on courser

훈련데이터셋에 359966의 다른 노래와 테스트데이터세세 224753의 다른 노래가 있다. 또한 59873의 노래가 테스트 셋에 있지만 훈련 셋에는 없다.(약 26.64%)

17/11/7

Xgboost(Extreme Gradient Boosting): 빠른 속도로 로직을 검증하기에 좋은 라이브러리

장점

1. 훌륭한 그레디언트 부스팅 라이브러리. 병렬 처리를 사용하기에 학습과 분류가 빠르다.
2. 유연성이 좋다. 평가 함수를 포함하여 다양한 커스텀 최적화 옵션을 제공한다.
3. 탐욕적 알고리즘을 사용한 자동 가지치기가 가능하다. 따라서 overfitting이 잘 일어나지 않는다.

\*overfitting: 과도하게 데이터에 대해 모델을 학습을 한 경우를 의미한다.

샘플데이터에 너무 정확하게 학습이 되었기 때문에, 샘플데이터에만 정확하고 정작 테스트 데이터에는 정확도가 떨어진다.

4. 다른 알고리즘과 연계 활용성이 좋다. Xgboost 분류기 결론부 아래에 다른 알고리즘을 붙여서 앙상블 학습이 가능한다. ResNet 마지막 바로 이전 단을 Feature layer로 응용하는 것과 비슷하다.

Boosting: 부스팅은 약한 분류기를 세트로 묶어서 정확도를 예측하는 기법이다. 모래 자갈, 먼지가 섞여 있는 물질에 여러 타입의 체를 가지고 조합해서 그것을 분류하는 과정과 유사.

예) 어떤 학습기 M에 대하여 Y를 예측할 확률은 아래와 같다.

Y=M(x)+error(1)

만약 error에 대하여 좀 더 상세히 분류할 수 있는 모델 G가 있다면 (단, error>error2)

error=G(x)+error(2)

로 표현할 수 있을 것이고, 여기에 error2를 더 세밀하게 분리할 수 있는 모델 H가 있다면(단 error2>error3)

Error2-H(x)+error(3)

으로 표현이 가능할 것이다. 따라서 이들을 모두 적용할 경우

Y=M(x)+G(x)+H(x)+error(3)->4

이는 학습기 M을 단독으로 사용했을 때보다 정확도가 높을 것이다. 하지만 4번 분류기는 개선의 여지가 있다. M,G,H 각각 분류기의 성능이 다른데, 모두 같은 비중(1\*M,1\*G,1\*H)을 두고 있기 때문에 임의의 x에 대하여 서로 간섭하며 오류를 높이는 결과를 낼 수 있기 때문이다. 이제 각 모델 앞에 비중(weight)을 두고, 기계 학습으로 최적의 비중을 찾자. 이것은 4번식의 모델보다 훨씬 더 좋은 성능(error3>error4)을 내는 분류기가 될 것이다.

Y=alpha\*M(X)+beta\*G(x)+gamma\*H(x)+error4->5

정리하자면, Xgboost는 탐욕적 알고리즘을 사용하여 분류기 M,G,H를 발견하고, 분산처리를 사용하여 빠른 속도로 적합한 비중의 파라미터를 찾는 알고리즘이다. 분류기는 Regression Score를 사용하여 정확도 스코어를 측정하고, 각 순서에 따라 강한 분류기부터 약한 분류기까지 랜덤하게 생성된다. 이렇게 만들어진 분류기를 트리(tree)라고 하며, 분류기를 조합한 최종 알고리즘을 포레스트라고 한다. 이것이 기본적인 boosting algorithm 원리이다.

LGBM

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks.

Since it is based on decision tree algorithms, it splits the tree leaf wise with the best fit whereas other boosting algorithms split the tree depth wise or level wise rather than leaf-wise. So when growing on the same leaf in Light GBM, the leaf-wise algorithm can reduce more loss than the level-wise algorithm and hence results in much better accuracy which can rarely be achieved by any of the existing boosting algorithms. Also, it is surprisingly very fast, hence the word ‘Light’.

Before is a diagrammatic representation by the makers of the Light GBM to explain the difference clearly

Leaf wise splits lead to increase in complexity and may lead to overfitting and it can be overcome by specifying another parameter max-depth which specifies the depth to which splitting will occur.

Below, we will see the steps to install Light GBM and run a model using it. We will be comparing the results with XGBOOST results to prove that you should take Light GBM in a ‘LIGHT MANNER’.

Let us look at some of the advantages of Light GBM.

배경지식

1. Learning Rate

Gradient Descent 알고리즘에서 Cost function이 반복을 거듭할수록 최소값으로 수렴해 최적의 해를 찾아가는 과정인데 Iteration을 수행할 때, 다음 point를 어느 정도로 옮길 지를 결정하는 것을 Learning rate라고 한다. 보통 알파로 정한다.

**Learning rate를 설정할 때 주의할 점은,**  
1) Learning rate가 너무 클 때,  
  최적의 값으로 수렴하지 않고, 발산해버리는 경우가 발생합니다. Overshooting이라고 하죠.  
2) Learning rate가 너무 작을 때,  
  일단 수렴하는 속도가 너무 느리고, local minimum에 빠질 확률이 증가합니다.

Regression(회귀 분석)

회귀 분석은 주어진 데이터가 어떤 함수로부터 생성됐는 가를 알아보는 함수 관계를 추측하는 것.

가설(방정식이) 몇 차 방정식인지? 그리고 계수는 각각 무엇인지 알아보는 것입니다.

이 방정식의 계수가 선형이면 선형 회귀분석(Linear Regression Analysis)

의사결정트리의 깊이와 넓이를 결정하는 방법

깊이를 너무 깊게하면 오버피팅이 일어난다.

앙상블: 머신러닝 모델을 연결하여 더 강력한 모델을 만드는 기법이다.

분류와 회귀 문제의 다양한 데이터 셋에서 효과적이다. 랜덤 포레스트와 그래디언트 부스팅 결정 트리는 둘다 모델을 구성하는 기본 요소로 결정 트리를 사용한다.

랜덤포레스트

랜덤포레스트는 분류, 회귀 분석 등에 사용되는 앙상블 학습 방법의 일종으로,

훈련 과정에서 구성한 다수의 결정 트리로부터 분류 또는 평균 예측치를 출력함으로써 동작한다.

랜덤 포레스트는 결정트리의 주요 단점인 오버피팅을 회피할 수 있게 한다. 여러 개의 결정트리의 평균을 냄으로써 오버피팅을 줄일 수 있다. 랜덤포레스트를 만들려면 트리를 만들 때 사용하는 데이터 포인트를 무작위로 선택하거나 분할 테스트에서 특성을 무작위로 선택하는 방법으로 나누어진다.

그래디언트 부스팅 회귀 트리

랜덤 포레스트와는 달리 그래디언트 부스팅은 이전 트리의 오차를 보완하는 방식으로 순차적으로 트리를 만든다. (무작위성이 없다) 대신 강력한 사전 가지치기가 사용된다. 그래디언트 부스팅에서 중요한 매개변수는 이전 트리의 오차를 얼마나 강하게 보정할 것인지를 제어하는 learning rate이다. 학습률이 크면 트리는 보정을 강하게 하기 때문에 복잡한 모델을 만듭니다. n\_estimators 값을 키우면 앙상블에 트리가 더 많이 추가되어 모델의 복잡도가 커지고 훈련 세트에서의 실수를 바로 잡을 기회가 더 많아집니다. 훈련세트 정확도가 100%이면 과대적합 과대적합을 막기 위해서 트리의 최대 깊이를 줄여 사전 가지치기를 강하게 하거나 학습률을 낮출 수 있습니다.