Predicting Future Sales with GBM

***손정현, 정진원**
서강대학교 컴퓨터공학과

Abstract

머신러닝 기술이 발전함에 따라 일상 속의 다양한 문제에 머신러닝을 적용할 수 있게 되었다. 상품의 판매량을 예측하는 문제 또한 머신러닝을 이용하여 해결할 수 있으며 본 보고서에서는 Kaggle: "Predict Future Sales" 컴페티션에서 주어진 데이터와 그래디언트 부스팅 기반의 머신러닝 기법인 LightGBM, XGBoost, CatBoost를 사용하여 모델별 판매량 예측 성능을 비교하였다. 단일 모델로서는 LightGBM의 예측 결과가 가장 우수하였으며, LightGBM과 CatBoost 두 모델의 예측 결과에 weighted average ensemble 기법을 사용했을 때 단일 모델보다 향상된 성능을 얻을 수 있었다.

I. 서론

본 보고서에서는 상품의 판매량을 트리 기반 그래디언트 부스팅 모델들을 사용하여 예측 성능을 비교 및 평가하고자 했으며, 피처 엔지니어링을 비롯한 다양한 기법을 이용하여 예측 성능을 향상하는 것을 목표로 하였다. Kaggle의 Predict Future Sales의 데이터를 이용하며, 2013년 1월부터 2015년 10월까지 상점별 상품 판매량을 바탕으로 2015년 11월 상점별 상품 판매량을 구하는 것을 세부 목표로 하였다.

연구를 진행하기 전에, Gradient Boosting 계열 모델인 LightGBM, XGBoost, CatBoost의 간략한 소개와 특징을 나열하고자 한다. 가장 먼저 등장한 XGBoost는 overfitting 을 방지하는 regularized term을 두고 second order taylor approximation을 사용하여 목적함수를 빠르게 도달할 수 있다. LightGBM은 XGBoost의 느린 속도를 보완하기 위해서 탄생하였다. XGBoost는 Information gain기반으로 split할 때, 모든 feature과 data에 대해서 접근해야하기 때문에 계산이 효율적이지 못하다. 따라서 LightGBM 큰 Gradient는 놔두고 작은 Gradient는 랜덤하게 드롭한 뒤, 일부를 샘플링해 더해주는 GOSS방식을 사용한다. 하지만 LIghtGBM은 데이터가 적을 경우 Overfitting할 가능성이 크다. CatBoost는 주로 Categorical 변수를 처리하는데 효과적인 알고리즘이다. 하지만 Sparse한 Matrix를 처리하는데 어려움을 겪고 특히 데이터 대부분이 수치형 변수인 경우, LightGBM보다 학습속도가 느리다.

현재 대회에서는 데이터가 굉장히 많고 Categorical 변수는 드물기 때문에 LightGBM 이 적합할 것으로 예상된다.

II. 데이터 분석 및 피처 엔지니어링

본 절에서는 주어진 네 가지 데이터: 카테고리(categories), 상품(items), 상점(shops), 판매정보(sales)를 시각화하여 분석하고, 도메인 지식을 바탕으로 주요한 피처들의 도출방식과 이유에 관해 서술한다.

2.1. 카테고리 데이터

카테고리 데이터는 상품들이 속할 카테고리들을 나타내며 표 1과 같이 구성되어 있으며 총 84개의 카테고리가 존재한다.

표 1. **카테고리 데이터의 형식**

category_name	category_id
PC - Headsets / Headphones	0
Accessories - PS2	1
Accessories - PS3	2
Accessories - PS4	3
⋮	⋮

category_name의 경우 대체로 '-'로 구분된 양상을 보여주며, 구분자의 앞부분을 추출하여 인코딩하여 그룹이라는 피처를 생성해주었다. 이는 카테고리 간의 연관성을 부여한다. 그림 1에서 확인 할 수 있듯이 Books 그룹과 기타 그룹에 가장 많은 카테고리가 포함되었다.

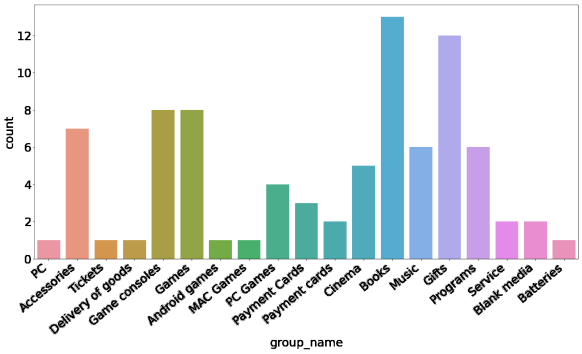

그림 1. 그룹별로 속해있는 카테고리의 개수

2.2. 상품 데이터

상품 데이터에는 상품이 속한 카테고리와 상품의 이름에 대한 정보를 준다. 총 22170개의 상품이 존재한다.

표 2. 상품 **데이터의 형식**

item_id	category_id	item_name
0	40	!! IN THE POWER OF HAPPINESS (PLAST) D
1	76	! ABBYY FineReader 12 Professional Edition Full [PC, Digital Version]
2	40	*** IN THE GLORY OF THE GLORY (UNV) D
3	40	*** BLUE WAVE (Univ) D
⋮	⋮	⋮

데이터를 분석한 결과 동일한 item_name에 대해 item_id가 중복으로 존재하는 경우가 존재하여 테스트셋에 존재하지 않는 id를 존재하는 id로 바꿔주었다. item_name의 경우 상품 간의 관계를 나타내주는 많은 정보를 가지고 있다고 판단하여 이를 피처로 추가해주었다. item_name의 길이가 비슷한 경우 유사한 상품인 경우가 많아 이름에서 특수문자를 제거하고 소문자로 만들어주는 전처리과정 전후의 길이를 피처로 추가해주었다. 추가로 전처리과정을 거친 item_name 간의 유사도를 Levenshtein Distance로 계산하여 60% 이상의 유사성이 있는 경우 같은 item_name_group으로 묶어주었다. item_name이 비슷한 경우 같은 item_name_group으로 묶인 것을 그림 2에서 확인할 수 있다.

	item_id	item_name	item_name_group
3001	3001	drweb security space rs pc 12 mont...	1063
3002	3002	drweb security space pc fault 12 m...	1063
3003	3003	drweb security space rs pc 24 mont...	1063
3004	3004	drweb security space rs pc 24 mont...	1063
3005	3005	drweb security space rs pc month...	1063

그림 2. item_name에 따른 item_name_group

2.3. 상점 데이터

상점 데이터에는 상점의 이름과 id가 주어진다. 총 60개의 상점이 있으며 표 3과 같은 형식으로 이루어져 있다.

표 3. **상점 데이터의 형식**

shop_name	shop_id
! Yakutsk Ordzhonikidze, 56 francs	0
! Yakutsk TC "Central" fran	1
Adygea TC "Mega"	2
Balashikha TC "Oktyabr-Kinomir"	3
⋮	⋮

상점도 중복된 이름이 존재하여 중복된 이름의 shop_id를 하나로 통일해주었다. 또한, shop_name 'Yakutsk TC "Central" fran'에서 볼 수 있듯이 shop_name에는 상점의 종류("Central")와 상점이 속해있는 도시(Yakutsk)에 대한 정보가 존재하여 이를 피처로 추가해주었다.

2.4. 판매 정보 데이터

판매 정보 데이터는 상품의 판매 날짜, 판매 상점, 판매 수량, 상품의 가격에 대한 정보를 준다.

표 4. 판매 정보 **데이터의 형식**

date	date_block_num	shop_id	item_id	item_price	item_cnt_day
02.01.2013	0	59	22154	999	1
03.01.2013	0	25	2552	899	1

05.01.2013	0	25	2552	899	-1
⋮	⋮	⋮	⋮	⋮	⋮

판매 정보 데이터의 경우 테스트 셋에 포함되지 않은 상점의 판매 정보를 포함하고 있다. 테스트하지 않는 상점에 대한 정보는 제거해주었다. 또한, 그림 3에서 볼 수는 item_price와 item_cnt_day에 존재하는 아웃라이어들을 제거해주었다. 주어진 날짜 정보를 이용하여 요일, 달, 년도, 수익, 아이템이 처음으로 팔린 날짜 등에 대한 feature를 추가해주었다.

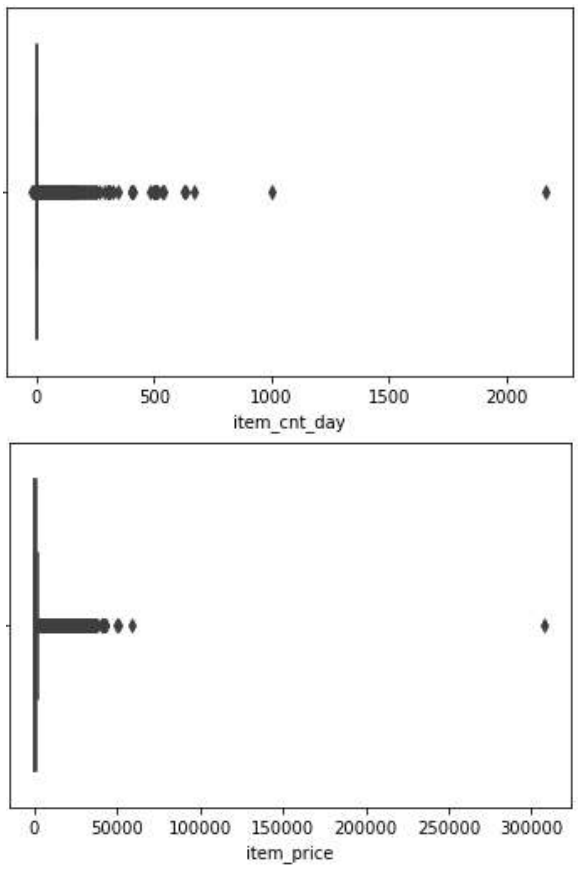

그림 3. item_cnt_day와 item_price의 아웃라이어

2.5. 추가 피처 생성

테스트 셋이 42개의 상점과 5100개의 상품에 대한 Cartesian product로 이루어져 있으므로 앞서 생성한 데이터들을 이용하여 월별로 판매 중인 상품과 상점에 대한 Cartesian product로 training 데이터프레임을 구성한다. 데이터프레임을 이용하여 추가적인 피처를 도출한다.

2.5.1 클러스터

클러스터링을 이용하여 상점과 카테고리들을 추가로 그룹 지었다. 상점과 카테고리는 판매 양상이 비슷하면 비슷한 그룹에 속한다고 생각하여 이를 바탕을 그룹을 결정하였으며 이는 PCA분석과 Agglomerative Clustering을 이용하여 진행하였다. PCA분석을 이용하여 상점과 카테고리에 대한 주성분의 개수를 정해주었고, component score에 따라 상점과 카테고리들을 평면상에 나타내주었다. PCA 분석을 통해 얻은 상점의 주성분 개수는 4개일 때 최적이었으며, 카테고리의 경우 2개일 때 최적이었다. 평면에 나타난 그림을 확인하여 클러스터의 개수를 상점과 카테고리 각각 5개와 4개로 정해준 뒤, Agglomerative Clustering 방식을 이용하여 클러스터링하였다. 결과는 그림 4와 5에서 확인할 수 있으며 이를 피처로 추가했다. 그림 4에서 shop_id가 55인 경우 다른 shop들과 멀리 떨어져 있는데 확인을 해보면 다른 상점들과 다르게 온라인 상품을 파는 것을 확인할 수 있다.

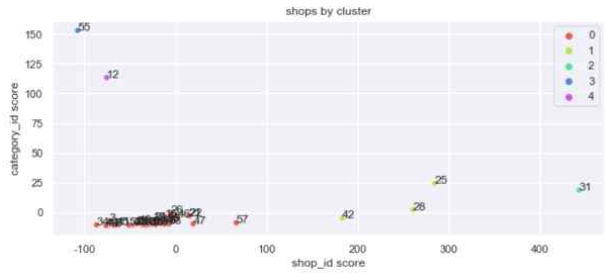

그림 4. 상점을 클러스터링 한 결과

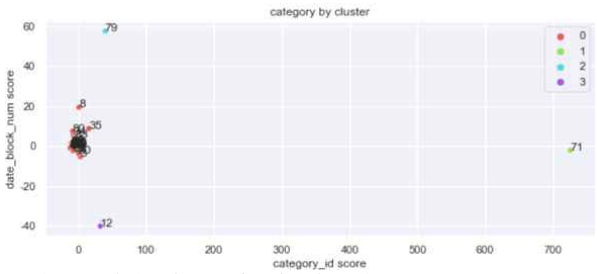

그림 5. 카테고리를 클러스터링한 결과

2.5.2 Age & Aggregated Features

Age feature은 각 feature마다 등장 이후로 얼마나 지났는지 보여주는 특징이다. Age feature을 사용한 이

유는 출시되었는지 얼마나 오래되었는지 따라서 판매량이 달라지기 때문에 이에 대처할 수 있다. 각 기준에 따라서 각 판매 달에서 가장 먼저 출시된 달을 빼서 age feature를 구하였다.

	date_block_num	item_id	item_age
0	0	27	0
7410	0	27	0
14820	0	27	0
22230	0	27	0
29640	0	27	0
...	...	...	...
4804039	17	27	17
4810267	17	27	17
4816495	17	27	17
4822723	17	27	17
4828951	17	27	17

그림 6. age_features를 추가한 결과

Aggregate features은 shop_id, shop_city등의 기준을 묶어서 평균 판매량을 구한다. 이를 통해 test 데이터에서 동일한 shop_id에 대해서 평균 판매량 정보를 얻을 수 있는 것이다.

2.5.3 Lag Features

Lag Features은 각 행의 판매월 기준으로 이전 달의 판매량입니다. 몇 개월 이전의 기록을 볼 것인지는 속도와 정확성을 이용해 현재부터 6개월 이전까지의 데이터를 기록하기로 결정하였다. 특히 12달 이전 달까지 기록하는 것은 시간도 너무 오래 걸렸지만 정확성에서 이전 6개월 이전의 기록을 살피는 것 보다 정확성이 좋지 못하였다. 또한, 단순히 이전 달들의 판매량을 살피는 것이 아니라 6개월전 판매량과 이전달의 판매량 사이의 변화량, 총합을 구하여서 모델이 lag와 관련된 다양한 특징을 참조할 수 있게끔 하였다.

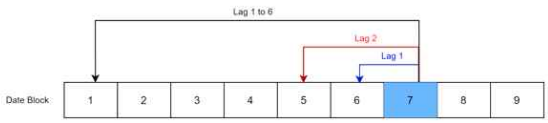

그림 7. lag_features를 표현한 그림.

2.5.4 new item

Validation set를 생성해서 분석해본 결과, item_age가 0인 제품들에 대해서 RMSE가 높은 것을 확인하였다.

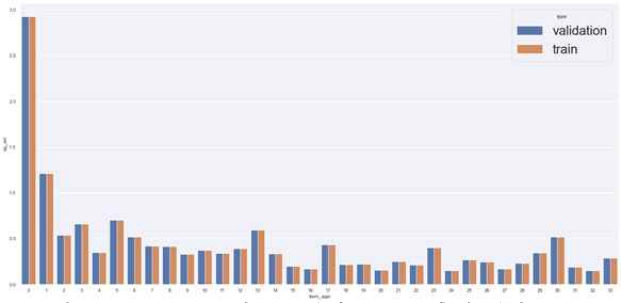

그림 8. item_age=0 인 item부터 RMSE 측정 결과

따라서 new item을 위한 feature인, category와 shop별로 새로운 아이템들의 평균 판매량인 new_items_in_cat 를 추가하였습니다. 결과적으로 itema_age가 0, 즉 새로운 아이템에 대한 RMSE가 2.9에서 2.8정도로 감소하였다는 것을 알 수 있었습니다.

	sq_err	target
new_item생선 전	2.927508	0.526227
new_item생성 후	2.800608	0.526227

표 5. new_items_in_cat 를 추가한 뒤 RMSE 결과

Ⅲ. 모델링 및 결과

3.1. 평가지표

모델의 학습 결과를 평가하는 방식은 RMSE(Root Mean Squared Error)를 이용하였다. RMSE는 모든 데이터에 대해서 예측값과 결과값을 빼고 제곱한 뒤, 총 개수로 나누고 그 결과에 제곱근을 취하여 계산한다. 이를 나타내면 다음 수식(1)로 나타낼 수 있다. 여기서 y_i는 참값 $\hat{y_i}$는 예측 값이다.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \qquad (1)$$

3.2. 모델 학습 및 검증

앞서 구축한 데이터를 이용하여 LightGBM, XGBoost, CatBoost 총 세 개의 모델을 학습시키고 예측을 하였다. LightGBM의 경우 Optuna를 이용한 hyperparmeter tuning을 진행하였다. 테스트 결과 표 5와 같이 LightGBM을 이용했을 때 RMSE가 가장 낮게 나와 성능이 가장 우수했고, XGBoost만을 사용했을 때 성능이 세 모델 중 가장 좋지 않았다.

표 6. 각 모델의 RMSE값

LightGBM	**0.86662**
XGBoost	0.89415
CatBoost	0.87650

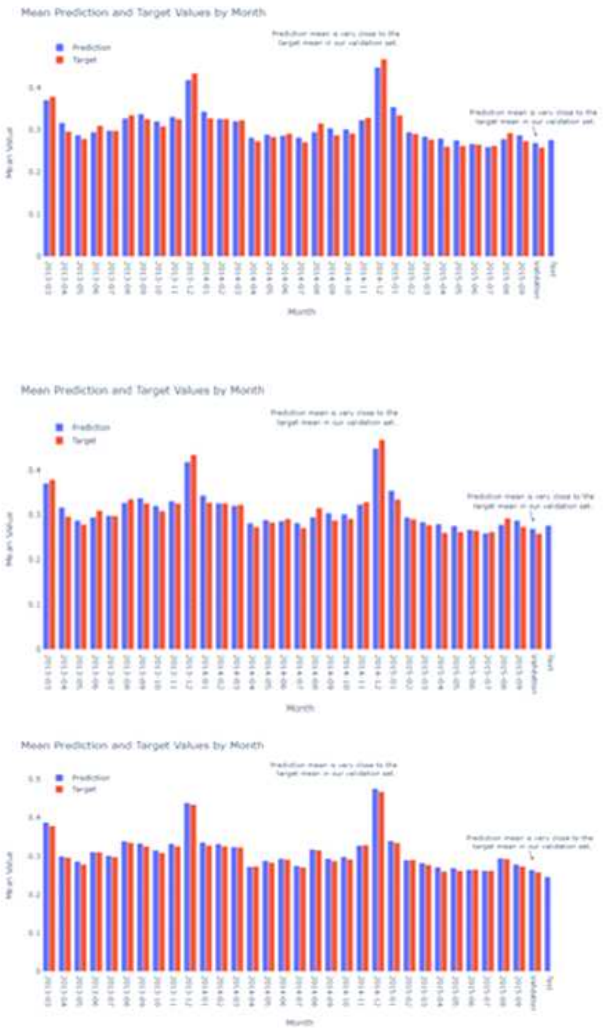

그림 9. 순차적으로 LightGBM, XGBoost, CatBoost의 월별 예측값과 실제 값을 비교한 그림

정확도를 향상시키기 위해 각 모델이 도출한 결과에 weighted average ensemble 방법을 적용하였다. 세 모델의 결과에 가중치를 다르게 부여하여 좋은 결과를 내는 가중치를 찾았으며, LightGBM과 CatBoost 두 모델의 결과에도 동일한 방식으로 가중치를 찾았다. 여러번의 시도를 통해 LightGBM, CatBoost, XGBoost 세 모델의 결과에 0.68, 0.224, 0.096의 가중치를 준 경우 0.85636의 RMSE를 얻어 단일 모델보다 향상된 결과를 얻을 수 있었으며, LightGBM과 CatBoost 두 모델에 0.6, 0.4의 가중치를 준 경우 RMSE가 0.85226으로 가장 우수한 성능을 보여주었다. 이는 6/17 오후 2시 Kaggle public score 기준 147/11835 등으로 약 상위 2%에 해당하는 결과이다.

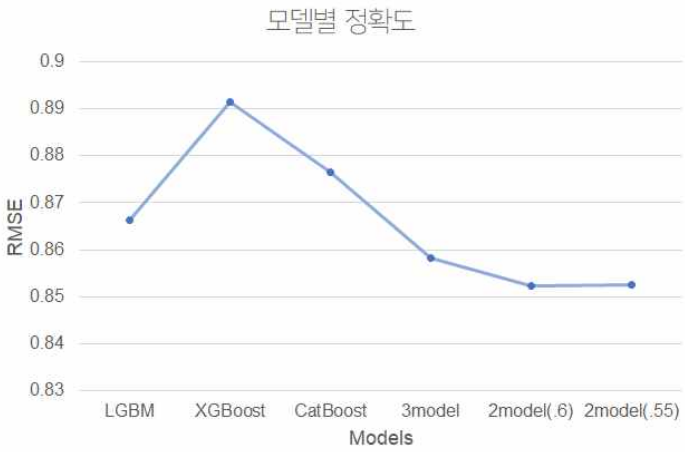

그림 10. 모델 별 정확도 2 model ensemble의 결과가 가장 우수하였다

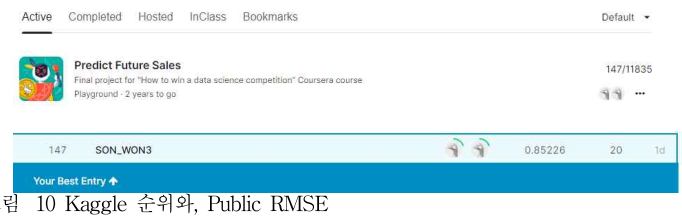

그림 10 Kaggle 순위와, Public RMSE

Ⅳ. 결론

본 글에서는 결측치 제거, PCA를 통한 클러스터링, 새로운 특징을 추가함으로써, 모델이 다양한 기준을 가지고 예측할 수 있도록 하였다. 모델로는 3가지 Gradient Boosting계열 알고리즘을 사용해서 정확도를 비교하였다. 예상과 동일하게 LightGBM이 가장 좋은 정확도를 가졌다. 이어서 weighted average ensemble 방법을 사용하여 최적의 가중치를 찾아내 좋은 결과를 이끌어냈다.

각 모델의 정확도를 측정하는 과정에서 XGBoost와 CatBoost에 대해서는 hyperparameter tuning을 하지 못한점이 아쉬웠다. 특히 XGBoost는 10%정도의 hyperparameter tuning을 수행하는데 24시간이 걸린만큼, 제한된 시간에서는 두 모델에 대해서는 이를 수행하기가 어려웠다. 또한, Post Processing을 다양하게 진행해봤지만 오히려 결과가 나빠졌다. 특히 RMSE가 높게 나왔던 새로운 상품과 상점에 대해서는 lag가 중요

한 feature가 아니기에 lag를 제거하고 새로운 상품을 잘 예측하도록 feature들을 추가하였지만 아쉽게도 결과적으로는 RMSE가 증가하였다.

V. 기여도

손정현(20161599)	정진원(20161643)
50%	50%