[2-2] Project3 Report

Status	시작 전

0️⃣ 팀 구성원 & 역할

- 김강산 : 데이터셋 탐색, 시계열 분석 담당
- 박유진 : 데이터셋 탐색, LSTM 담당
- 조민수 : 데이터셋 탐색, 칼럼별 분석

1️⃣ 활용 데이터 선정 및 수집

- 활용한 데이터 셋

상점 신용카드 매출 예측 경진대회

출처 : DACON - Data Science Competition

https://dacon.io/competitions/official/140472/overview/description

store_id	상점 고유 아이디
card_id	카드 고유 아이디
card_company	비식별화된 카드 회사
transacted_date	거래 날짜
transacted_time	거래 시간 (시:분)
installment_term	할부 개월 수(포인트 사용시(60개월+실제 할부개월)을 할부 개월 수에 기재)
region	상점의 지역
type_of_business	상점의 업종
amount	거래액 (단위는 원이 아님)

2️⃣ 데이터 분석 및 실제 학습 데이터, 검증 데이터 선정

👉 해당 섹션은 아래와 같이 나눠져 있음
1. Column Analysis을 위한 데이터 처리
2. Time series(ARIMA Model)을 위한 데이터 처리
3. LSTM 모델링을 위한 데이터 처리

1. Column Analysis 을 위한 데이터 처리

- 칼럼 선택
  - ‘region’ , ‘type_of_business’ 칼럼을 기준으로 분석을 진행하기로 결정
    - region 칼럼에 데이터가 존재하는 데이터프레임 : train_region
    - type_of_business “ : train_business
    - 두 칼럼 전부 “ : train_all
- ‘amount’ 음수값 처리’
  - train_region → trian_region_remove / train_business → train_business_remove / train_all → train_all_remove
- region칼럼 분석
  - 지역별 매출
    - 행정구역마다 데이터를 나누어 행정구역(시)마다 매출(amount)확인.
  - 지역별 카드회사 분포
    - 데이터프레임의 분리가 필요하다고 생각하여 행정구역별로 데이터프레임 분리 후 저장
    - 행정구역마다 카드회사 분포 확인 [region_card_df]
  - 지역별 비즈니스
    - 총 145개의 비즈니스가 있었고 처음에는 모든 칼럼을 가져갔으나, 멘토님과의 상담 이후 공통점이 보이는 칼럼은 서로 묶어서 10개의 칼럼으로 줄였다.
    - 행정구역마다의 비즈니스 분포를 추후 활용하기 위해 따로 저장 [region_business_df]
  - 지역별 NaN 매출

        - 지역-비즈니스가 존재하는 train_all_remove를 train데이터로 학습시키고,
        - train_region_remove 중 지역은 존재하되 비즈니스가 NaN값인 데이터를 이용해 해당 비즈니스가 어떤 비즈니스인지 예측해보는 모델을 만들려고 했다.
            - 경기/서울 지역의 데이터 밀집화, 너무 많은 NaN값(214만)과 칼럼 등 여러가지 실패 >> 추후 개선 필요
                - 추후 지역을 한번더 묶거나, store_id를 통해 새로운 데이터프레임으로 분리해서 찾아보거나, 이미 존재하는 데이터값(12만)로 모델을 만들거나 해서 보완하도록 하겠다,
- business칼럼 분석
    - 비즈니스별 매출
        - 위의 '지역별 매출'과 동일하게 모든 비즈니스에 대해 매출 확인
    - 비즈니스별 카드회사 분포
        - 각 비즈니스에 대해서 정리를 하고 싶어, 변수-비즈니스명 관계를 가진 딕셔너리를 생성했고, 앞으로 사용할 변수가 어떤 비즈니스인지 확인하는 용도이다.
        - 각 비즈니스별로 데이터를 나누어 특정 비즈니스의 특징과 분포를 확인
        - 비즈니스마다 카드회사 분포 확인 [business_card_df]
    - 비즈니스별 store_id
        - 추후 store_id도 분석에 활용하기 위해 미리 비즈니스-store_id 관계를 나타내는 데이터프레임이 필요하다고 생각했다.
        - 비즈니스마다 store_id 분포 확인 [business_store_id_df]
- 두 칼럼 분석
    - store_id별 비즈니스
        - 위에서 진행한 비즈니스별 store_id의 역순으로 각 store_id가 무슨 비즈니스인지 정리한 딕셔너리를 생성

2. Time Series Analysis를 위한 데이터 처리

시계열 분석을 위한 전처리 진행

Time → Day Resampling

> 기존 데이터에선 시간 단위로 데이터를 제공하기에 이를 일 단위 데이터로 resampling을 했음

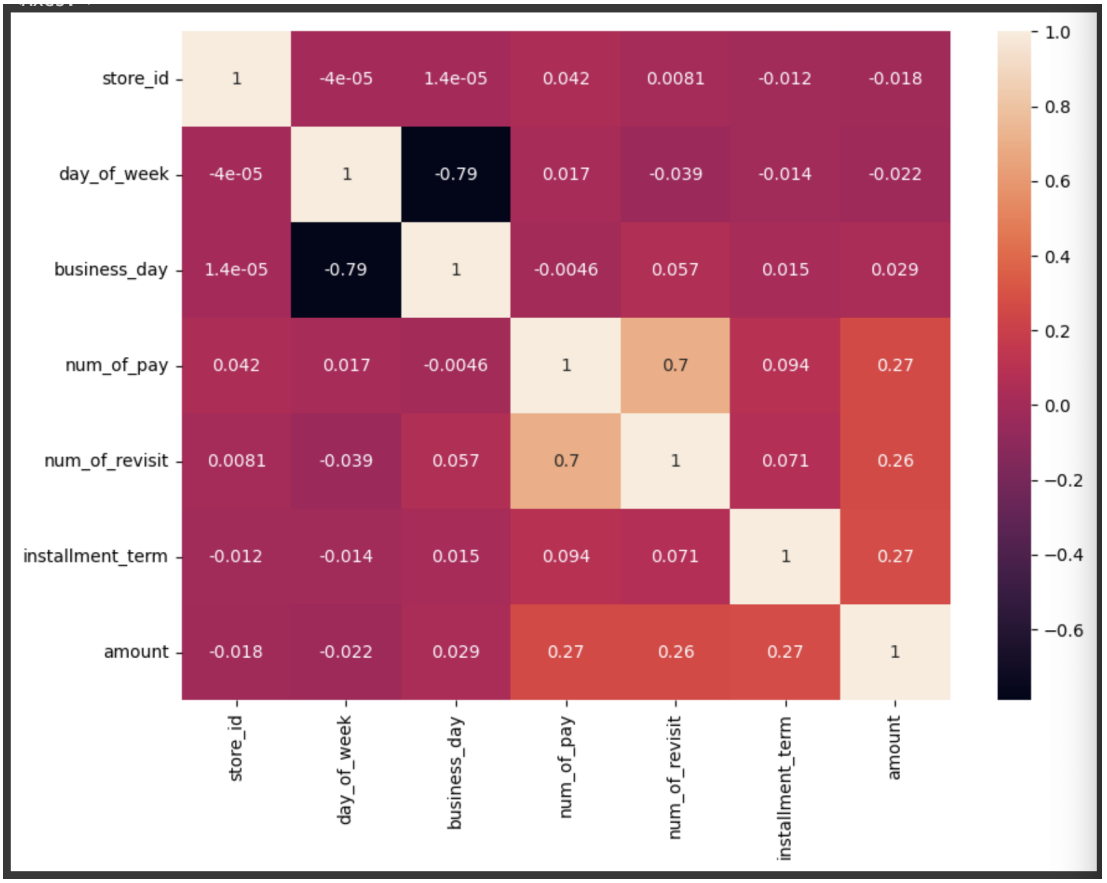
- 시간 단위 데이터

transacted_date	store_id	card_id	card_company	transacted_time	installment_term	region	type_of_business	amount
2016-06-01	0	0	b	13:13	0	NaN	기타 미용업	1857.142857
2016-06-01	0	1	h	18:12	0	NaN	기타 미용업	857.142857
2016-06-01	0	2	c	18:52	0	NaN	기타 미용업	2000.000000

- 일 단위 데이터

transacted_date	store_id	day_of_week	business_day	num_of_pay	num_of_revisit	installment_term	region	type_of_business	amount
2016-06-01	0	2	1	4	4.0	0	NaN	기타 미용업	12571.428571
2016-06-02	0	3	1	7	3.0	0	NaN	기타 미용업	40571.428571
2016-06-03	0	4	1	3	2.0	0	NaN	기타 미용업	18142.857143

- 일 단위 데이터로 파악한 컬럼간 상관관계 파악



Day → Month Resampling

> 조금 더 거시적인 변화방향을 확인하기 위해 일단위 데이터를 월단위 데이터로 Resampling

- 일 단위 data

transacted_date	store_id	day_of_week	business_day	num_of_pay	num_of_revisit	installment_term	region	type_of_business	amount
2016-06-01	0	2	1	4	4.0	0	NaN	기타 미용업	12571.428571
2016-06-02	0	3	1	7	3.0	0	NaN	기타 미용업	40571.428571
2016-06-03	0	4	1	3	2.0	0	NaN	기타 미용업	18142.857143

- 월 단위 data

transacted_date	store_id	real_tot_day	real_business_day	num_of_pay	num_of_revisit	installment_term	amount
2016-06-30	0	25	17.0	145.0	77.0	13.0	7.470000e+05
2016-07-31	0	26	16.0	178.0	105.0	24.0	1.005000e+06
2016-08-31	0	24	16.0	171.0	97.0	69.0	8.715714e+05
2016-09-30	0	25	19.0	160.0	103.0	15.0	8.978571e+05
2016-10-31	0	26	16.0	167.0	115.0	9.0	8.354286e+05

3. LSTM을 위한 데이터 처리

1. **결측치 제거**

- ‘region’, ‘type_of_business’의 경우 결측치가 많아 삭제하고 다른 파생변수를 생성해보기로 결정

2. **중복값 제거**

- 동일 일자, 카드 회사, 카드 아이디, 거래액 등을 가진 중복값을 파악하고 제거

3. **음수값 처리**

- 동일 카드 회사, 아이디, 거래액을 가진 거래액이 양수, 음수로 매칭이 되는 경우 환불로 간주하고 삭제하는 처리를 진행
- 만약 매칭이 되지 않음에도 음수로 나오는 경우에는 drop

4. **변수 추가**

- 하루 총 고객 방문 횟수를 ‘total_daily_visits’로 추가함

5. **데이터 resampling**

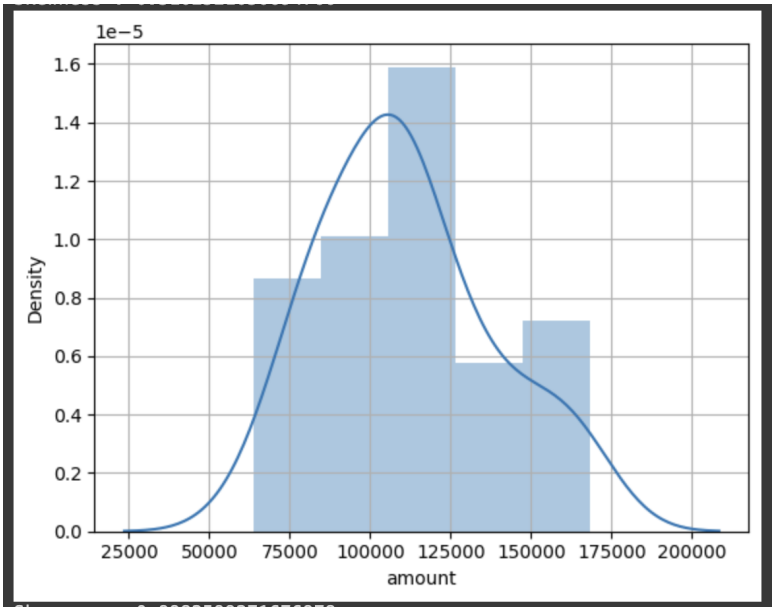
- 예측해야하는 값은 총 3개월 amount값으로, 현재 일자, 더 세부적으로 시간 별로 나뉘어져 있기에 월 단위로 resampling을 진행함

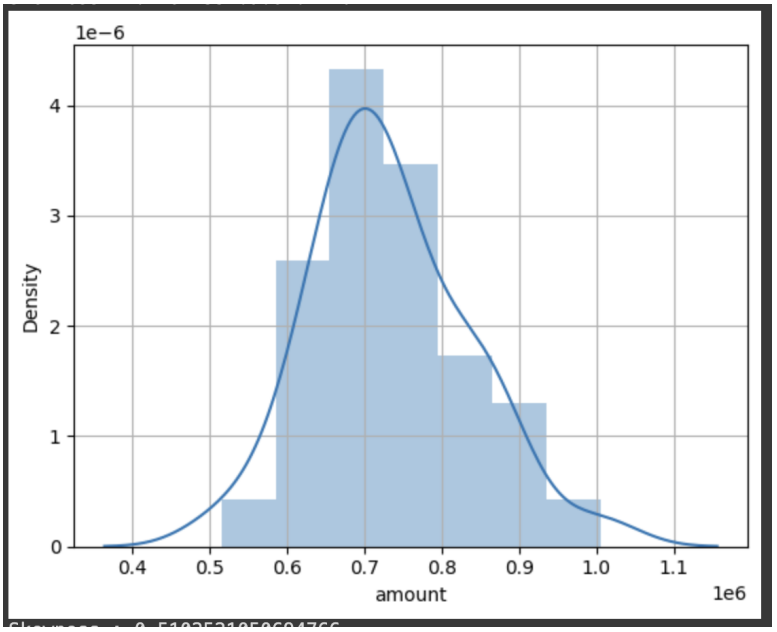
	store_id	transacted_date	installment_term	amount	total_daily_visits
0	0	2016-06-30	13	7.555714e+05	4
1	0	2016-07-31	24	1.005000e+06	9
2	0	2016-08-31	69	8.715714e+05	6
3	0	2016-09-30	15	8.960000e+05	4
4	0	2016-10-31	9	8.374286e+05	10
...	...	...	...	...	...
60227	2136	2018-10-31	0	2.026857e+06	8
60228	2136	2018-11-30	0	2.162714e+06	11
60229	2136	2018-12-31	0	2.452500e+06	22
60230	2136	2019-01-31	3	1.873643e+06	11
60231	2136	2019-02-28	0	2.266429e+06	15

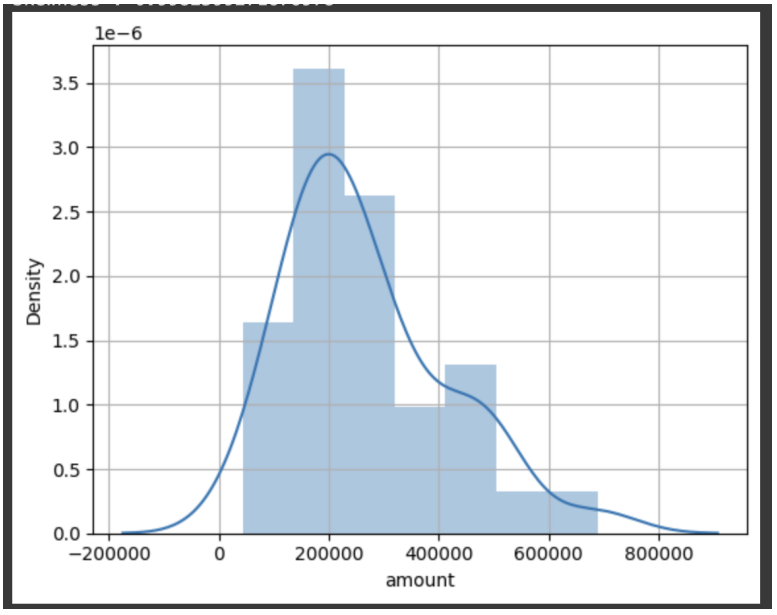
60232 rows × 5 columns

3️⃣ 예측 모델 구현을 위한 학습 진행

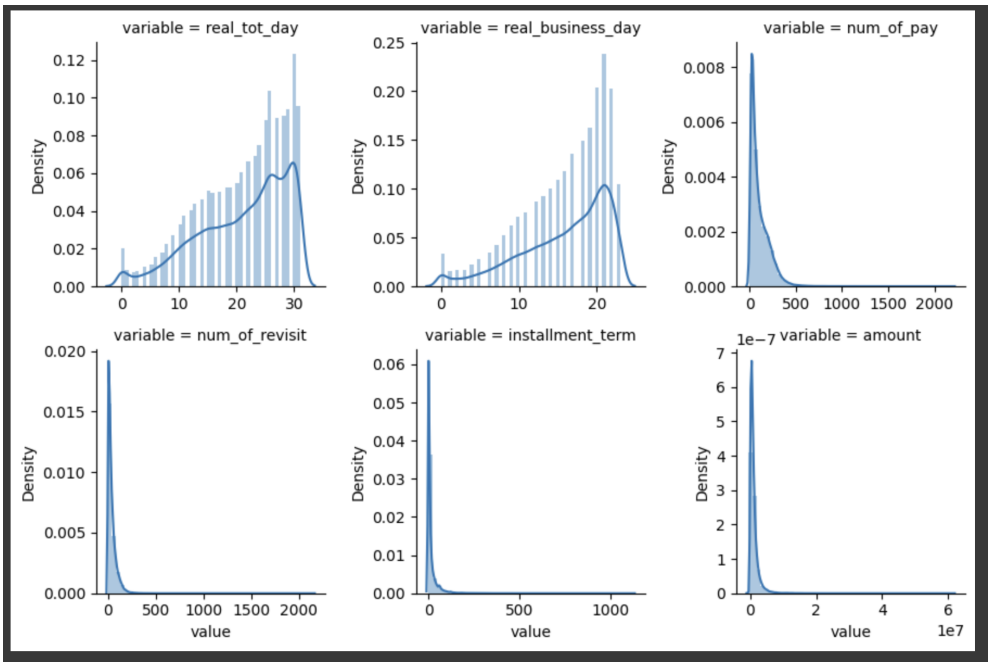
1. Time Series, ARIMA Model
    a. ARIMA Model이란?
        i. 기존 AR, MA, ARMA 모델 데이터가 stationary해야 분석이 용이했던 것에 반해 Non-stationary한 경우 차분을 통해 데이터를 정상으로 변형해 주어야 한다. ARIMA는 ARMA 모형에 차분을 d회 수행해준 모델을 뜻한다.
    b. 데이터 정규성 파악


1e−5
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Density
25000
50000
75000
100000
125000
150000
175000
200000
amount

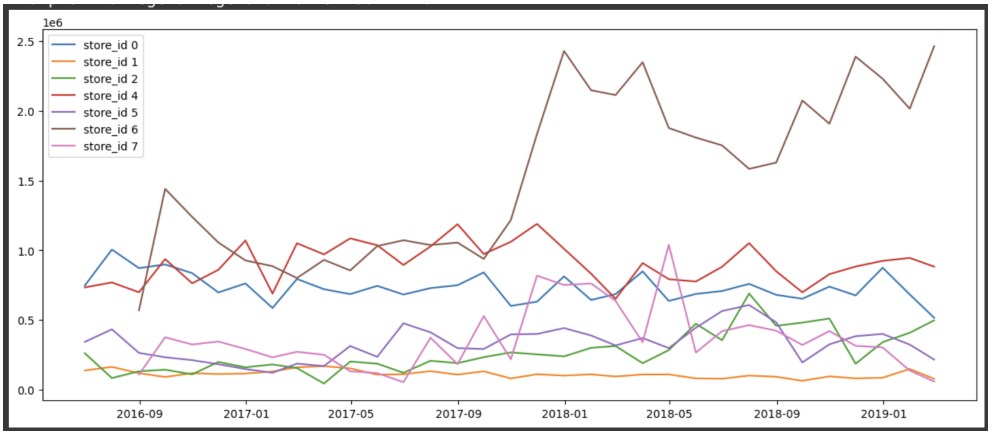

1e−6
4
3
2
1
0
Density
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
amount
1e6



c. value 분포 시각화



d. 여러 상점 매출 추이 확인



e. Modeling

i. 파라미터 설정

```
p = list(range(0, 6))
d = [0, 1, 2]
q = list(range(0, 6))

pdq = list(itertools.product(p, d, q))
```

ii. 모델 학습을 위한 코드

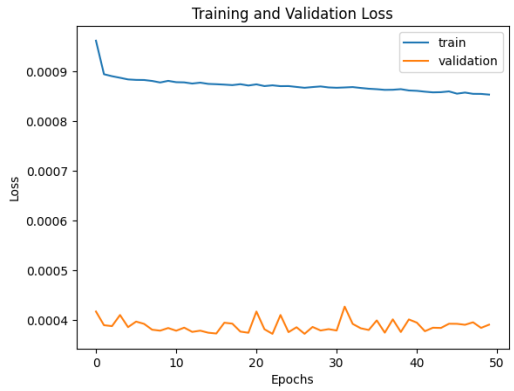
```
df_set = df_month[df_month.store_id == 0]

size = int(len(df_set) * 0.7)
train = df_set[:size]
test = df_set[size:]

best_score = 10000000
best_param = 0
for param in pdq:
    try:
        arima_model = ARIMA(train.amount.values, order=param)
        result = arima_model.fit()
        if result.aic < best_score:
            best_score = result.aic
            best_param = param
    except:
        continue

set_arima = ARIMA(df_set.amount.values, order=best_param)
set_result = set_arima.fit()
set_pred = set_result.forecast(len(test))[0]
```

2. **LSTM**



- train set에 대해서는 epoch이 증가할 수록 loss값이 작아지나, validation에서는 간간히 튀어오르는 값이 존재
- 다만 MSE가 0.00031010489328764 38로, 낮은 값이라고 판단

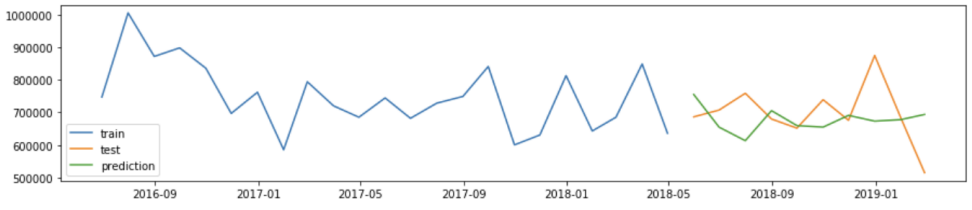
→ 이에 store_id별로 총 3개월 amount 예측을 진행해보기로 결정

- 과적합 방지를 위해 Dropout 레이어를 사용
- 시퀀스 길이 = 3 (분기별 데이터를 사용, 초기 7에서 변경)
- ReLU를 활성화 함수로 사용 (거래액 예측에는 linear를 사용하기도 하나, amount가 음수가 되면 안 되고, ReLU는 음수 입력에 대해 0을 출력하여 해당 활성화 함수를 채택)
- validation_split 값을 설정한 경우, 예측이 진행되는 도중 에러가 발생하여 설정하지 않고 진행

4️⃣ 구현된 가격 예측 모델 결과 확인

1. ARIMA Model
    a. 결과값

```
Mean value of store_id 0 : 732559.7402597402
AIC Score of test : 557.6623843461919
Best parameter of (p, d, q):  (2, 2, 1)
```



    b. ARIMA Score : 1080182.482790
        i. ARMA Score : 984368.752690

→ ARMA Score에 비해 낮은 예측치를 보임

2. **LSTM**

- Score : 963023.7370090218

5️⃣ 예측 모델의 성능 평가 (이미지 첨부)

1. ARIMA Model
    a. 시계열 데이터 분석에 있어서 ARIMA 모델은 기존의 ARMA 모델이 과거의 데이터만 분석에 사용하는 것에 반해 과거의 데이터가 지니고 있는 추세까지 분석에 반영한다.
    b. 하지만 ARIMA Model의 예측도가 떨어지는 것은 추세의 일관성이나 유의미성이 크게 나타나지 않아 정확도가 떨어졌다고 판단할 수 있다.
2. LSTM
    a. LSTM 모델은 직전의 데이터 뿐만 아니라 과거의 데이터를 이용하여 미래를 예측한다는 점에서 거래액 예측에 용이할 것이라 판단했었다.
    b. 다만 score 비교를 위해 음수값을 0으로 전처리하였던 시계열 모델을 제출하고 score를 비교해본 결과, LSTM보다 예측도가 좋았다. 이에 LSTM 모델이 과거 데이터를 이용하더라도 무조건 좋은 모델이라고 판단할 수 없다.
    c. 이는 전처리의 영향을 받았을 것이라 판단하며, 시퀀스 길이를 조금 더 길게 설정하는 등 조정을 하는 경우 정확도가 더 높아질 수도 있을 것 같다.

6️⃣ 멘토 피드백

Column Analysis

- 머신러닝을 진행하려 했던 데이터 (region값은 있지만 type_of_business값은 없는 데이터)에 대해 피드백 요청한 결과
    1. 전체적으로 칼럼 수가 많아서 정확한 분석이 안나올 것이므로 칼럼을 통합시켜보라고 하셨습니다.
    2. groupby를 store_id에 적용해서 시계열 분석이든 통계적 분석이든 진행하라고 하셨습니다.

Time Series Analysis + ARIMA Model

- 시계열 분석은 머신러닝과 다른 맥락을 가지고 있다. 실제 현업에서도 시계열 분석만 담당하는 사람들이 있다.
- 앞으로 행동 데이터, 유저 로그 데이터 등을 분석할 때 시계열과 이상치 탐지등을 접할 수 있을 것이다.
- 모델의 개선 방향이 필요하다. 일정한 프로세스를 정해하고 그 프로세스를 수없이 반복하며 어떻게 개선했는지 분석하고 보고했으면 좋겠다.

LSTM

- 딥러닝 분야는 모델을 이해하는 것도 중요하지만, 에러가 발생한 경우 어떻게 처리할지 등 학습 환경에 대한 공부 역시 필요하다.
- 우선 score는 확인할 수 있도록 진행해야한다.

7️⃣ 프로젝트 결과

분석 대상은 신용카드 매출액 데이터로, 시간에 따른 매출액의 패턴을 파악하고 예측하는 것이 주요 목표였다. 이를 위해 ARIMA(ARIMA(p,d,q)) 모델을 활용하여 시계열 데이터를 분석했다. ARIMA 모델 성능 평가에 있어서 진행은 데이터를 훈련 및 평가 데이터로 세분화 하고 ARIMA 모델을 통해 예측을 수행했다. 예측 성능을 측정하기 위해 평균 절대 오차 (MEA, Mean Absolute Error) 등의 지표를 사용했다. ARIMA와 ARMA 모델을 비교했을 때, ARIMA 모델 예측 성능이 ARMA 모델보다 낮게 나타났다. 이에 대한 원인은 추세의 일관성이나 유의미성이 크게 나타나지 않아 정확도가 낮게 나타났다고 판단된다. 만약 ARIMA 모델을 계속 사용하여 해당 모델의 성능을 더 좋게 만들기 위해선 ARIMA 모델에 추가적인 요소를 추가하여 모델의 복잡성을 높이고 예측 성능을 향상시키는 방안과, 하이퍼파라미터 튜닝을 조정하면서 최적의 조합을 찾아보는 방안들을 생각해 보았다.

ARIMA 모델을 통한 시계열 분석은 유의미한 결과를 도출하였으나, ARIMA 모델의 한계를 인지하고 개선의 여지가 있음을 확인하였다. 향후 모델은 추가적인 feature engineering, 다른 시계열 모델의 탐색, 외부 요인의 고려 등을 통해 모델의 성능을 더욱 향상시킬 수 있도록 하겠다.

LSTM 모델의 경우에도 성능이 좋게 나타나지는 않았다. train-validation set으로 분할하여 학습하였을 때는 mse가 0.0003 수준으로 낮아 실제 학습도 무난히 진행할 것이라 판단했다. 그러나 실제로는 다른 시계열 모델보다도 score가 좋지 않은

경우가 있었고, 이는 학습 시간 등을 고려하여 시퀀스 길이를 짧게 조정하였던 것이 과거 데이터를 기반으로 미래를 예측하는 부분에 한계로 작용했기 때문인 것 같다. 이에 추후 LSTM 모델을 사용하게 된다면, 전처리나 적절한 파라미터 설정 및 학습 환경에 대한 고려를 통해 성능을 향상시키도록 하겠다.

8️⃣ 프로젝트 진행 후 느낀점

- 김강산 : 시계열 분석을 접했을 때, 머신러닝 보단 통계에 더 가깝다는 글을 보았다. 실제로 많은 부분에 있어서 통계적 지식을 요구했고 이를 이해하고 따라가기 위한 노력이 많이 수반되었다. 특히 통계적인 개념을 코딩으로 어떻게 구현했는가를 초점으로 두고 따라가고자 노력했다. 또한 시계열 데이터를 분석하기 위한 더 좋은 모델을 찾기 위해 노력했으며, 조금 더 복잡한 모델이 항상 더 나은 성능을 내지 못한다는 것은 흥미롭게 여겨졌다.
- 박유진 : 모델에 집중하느라 앞선 과정에 집중하지 못한 점이 아쉽다. 여러 인사이트를 좀 더 찾아볼 수 있었을 것 같은데 그러지 못한 점이 아쉬웠다. 또, 새로운 모델을 알아가는 것이 어려우면서도 재밌었다. 다른 프로젝트에서 잠깐 써본 경험이 있어서 조금 더 내가 모델에 대해 알고 싶어 담당하겠다고 했지만, 아직 어렵고 시간이 걸리는 일이란 걸 알았다. 그래도 그동안 어려워하던 시계열 데이터를 시도했다는 것만으로도 유의미했다고 생각하고, 프로젝트가 끝나도 개인적으로 공부하고 다른 모델을 시도해보려고 한다.
- 조민수 : 다른 인사이트를 찾기 위해 다른 관점에서 분석을 진행해봤다. 분석을 진행하면서 특별한 인사이트를 찾기가 힘들 뿐더러, 내가 원하는 내용을 얻기 위해 데이터프레임을 어떻게 설계,분리해야하고, 어떤 전처리 과정을 거쳐야하는지 알아내는데 시간이 걸렸다. 하지만 모든 과정이 끝내고 내가 분석하기 편한 데이터프레임을 만들어 놓으면 그 뒤 분석은 꼬리에 꼬리를 물듯 유연하게 흘러갔다. 하나의 주제를 분석하다가 새로운 아이디어가 떠올라 또 분석하고 .. 나중 가서는 기존 주제와 섞여 복잡하긴했지만, 매번 새로운 인사이트를 찾아내는 과정이 재미있었고, 시간이 촉박하여 마저 분석하지 못했던 아이디어를 추후에 다시 해보도록 하겠다.