

# Finda앱 데이터를 통한 대출신청 예측



By. FiveGuys

곽수민 (ytnals3803@naver.com)

김민석 (ipad39@cau.ac.kr)

윤우성 (dntjd0804@naver.com)

이승재 (zzzsss280@naver.com)

정석범 (abc07101@naver.com)


# 목차 A table of contents.

---

- 01 데이터 전처리 방법
- 02 활용 알고리즘 설명
- 03 최종 예측 결과 및 타당성
- 04 서비스 메시지 제안







# Part 1.

## 데이터 전처리 방법

## 새로운 변수 생성

- 001 >> 주어진 데이터 뿐만이 아닌 이를 파생하여 새로운 변수로 만든다면 기존의 데이터 보다 더 좋은 모델 성능 향상을 띄었기에 새로운 변수들을 생성하였다.
- 새롭게 생성한 변수로는 **Age**, **limit\_disired**이며, **Age** 변수의 경우 현재년도에서 출생년도를 빼서 만들었고, **limit\_disired** 변수의 경우 개인의 대출희망금액과 대출한도금액의 차이로 계산하여 만들었다.
- 

## 연속형 자료 스케일링

- 002 >> 변수 중 **yearly\_income**과 새로 만든 변수 **difference\_loan**는 모두 단위가 원 단위로 해당 데이터들의 범위가 컸으며 일부 고소득자들에 의하여 정확한 모델 학습이 어려웠다.
- 특히 연소득(**yearly\_income**)의 경우 변수에 **StandardScaler**를 통해 표준화를 진행하여 특정 데이터가 편향성을 갖는 것을 방지 하였다.

## 변수 제거

- 003 >> `User_spec` 데이터와 `loan_result` 데이터를 일단 합해준 후 이 중 대출 신청에 연관이 없다고 여겨지는 변수들을 제외시켰다.
- `application_id`, `user_id`, `insert_time`, `bank_id`, `company_enter_month`는 대출 예측에 끼치는 영향이 적었기에 제외하였고, `birth_year`, `loan_limit`, `desired_amount`는 새로운 변수 생성 후 삭제하였다.
- 

## 라벨 인코딩

- 004 >> 범주형 데이터가 많아 이를 모두 더미 변수로 만들면 차원이 기하급수적으로 커져 처리시간이 증가하였다. 우리의 분석결과 트리 기반 모델들의 성능이 높았기에 트리 기반 모델을 사용하기로 했다. 특히, 트리 모델의 경우 다른 수학적 모델들과 달리 범주형 데이터를 더미변수화 하지 않아도 되기 때문에, 더미변수화 된 데이터와 비교했을 때 성능상 큰 차이가 나지 않았고 차원을 절약할 수 있어 빠른 학습이 가능했다.

## 결측치 대체

- 005 >> User\_spec과 loan\_result 데이터를 application\_id 컬럼을 기준으로 합친 후, 결측치의 개수를 확인하였다. 분석에 사용하지 않거나, 결측치 대체 이외의 방법을 사용할 컬럼인 birth\_year, gender, company\_enter\_month 를 제외한 나머지 변수에 대해 결측치 대체를 진행했다. 이 과정에서 IterativeImputer를 사용하였다. IterativeImputer는 Round-robin 형식을 기반으로 결측치 대체를 수행하는 회귀 알고리즘인데, 결측치가 존재하는 변수들이 다른 변수들에 의해 영향을 받는다고 판단하였기 때문에 임의로 대체하기보단 예측을 하는게 적합하다고 판단하여 사용하였다.

application_id	0
user_id	0
birth_year	128038
gender	128038
insert_time	0
credit_score	1508618
yearly_income	6
income_type	0
company_enter_month	399201
employment_type	0
houseown_type	0
desired_amount	0
purpose	0
personal_rehabilitation_yn	5885910
personal_rehabilitation_complete_yn	11787488
existing_loan_cnt	2684723
existing_loan_amt	3888718
loanapply_insert_time	0
bank_id	0
product_id	0
loan_limit	0
loan_rate	0

## 시간대별 구분

- 006 >> 주어진 데이터 중 log 관련 데이터와 대출신청시간(`loanapply_insert_time`) 과의 연관성이 있다는 분석결과를 얻었다. 그래서 시간대별로 대출한도(`loan_limit`)의 차이를 확인하고자 24시간을 각각 6시간으로 나눠 시간 그룹별로 대출한도(`loan_limit`)의 차이가 있는지 판단하였다. 그 결과 6~12시와 12~18시에 사용한 사용자의 대출한도가 같다는 정보 이외에 나머지 그룹별로 차이가 있다고 확인되어 그룹화한 변수를 사용하였다.

Multiple Comparison of Means - Tukey HSD, FWER=0.05						
group1	group2	meandiff	p-adj	lower	upper	reject
1	2	493300.4283	0.0	280551.364	706049.4925	True
1	3	594560.6868	0.0	385823.5787	803297.7949	True
1	4	1240418.9549	0.0	1021525.0421	1459312.8677	True
2	3	101260.2586	0.1409	-20367.8685	222888.3857	False
2	4	747118.5266	0.0	608782.826	885454.2273	True
3	4	645858.268	0.0	513775.7861	777940.75	True



A person wearing a white lab coat and dark shoes is walking on a bright yellow surface. A long, dark shadow is cast to the left of the person. The person is carrying a black bag. The background is a solid yellow color with some faint lines.

## Part 2.

## 활용 알고리즘 설명



## Part 2, 활용 알고리즘 설명



# CatBoost

A

기존 부스팅 모델들의 단점을 개선한 모델로 특히 부스팅 모델의 가장 큰 단점인 느린 학습속도와 **Overfitting** 문제를 개선하였다.

B

데이터의 순서를 섞어주는 방법을 통해 모델이 같은 순서대로 잔차를 반복 예측할 가능성을 최소화하였고 범주형 피처의 인코딩시 **Dark Leakage** 문제를 방지하는 기법인 **Random, permutation, Overfitting detector** 등을 사용하여 오버피팅을 최소화한다.

C

본 팀의 전처리 데이터 셋에 범주형 변수가 특히 많았다. 여러 모델의 성능을 비교해본 결과 **CatBoost**가 성능면에서 가장 우월하였고 학습 속도 또한 준수하게 나타났기 때문에 **CatBoost**를 메인 모델로 선정하였다.

## CatBoost모델 적용

## STEP 1

전처리한 데이터셋을  
**Under Sampling**을 통해  
클래스 불균형을 해소해  
준다.

&gt;&gt;

## STEP 2

GridSearChCV를 통해  
모델의 최적값을  
찾고자 하였다. 이를  
통해 하이퍼파라미터  
값으로 **learning**  
**rate=0.5,**  
**max\_depth=5,**  
**n\_estimators**는  
**2000**이라는 값을  
획득했다.

&gt;&gt;

## STEP 3

해당 하이퍼 파라미터로  
**CatBoost Classifier**를  
사용한 결과 **validation**  
**set**에 대해 **f1 score**는  
**0.35** 를 얻었다.

## Part 2, 활용 알고리즘 설명

# K-MEANS Clustering

A

각 데이터로부터 클러스터들의 중심까지의 유클리드 거리를 계산한 후, 센트로이드를 변경하며 반복을 진행한다.

B

유클리드 공간과 클러스터의 수에 따라 NP-난해를 이용해 최적의 값을 찾는다. 주로 휴리스틱 기법을 사용하며 대규모 데이터 처리에 용이한 장점을 가진다.

C

숨겨진 패턴을 찾아 유사한 군집끼리 묶어야 하기에 분류의 목적이 아닌 군집화를 목적으로 설정했다. 따라서 비지도 학습 방법인 **K-Means**가 적절하다고 판단하여 사용하였다.



## K-Means모델 적용

## STEP 1

전처리한 데이터셋을  
**Under Sampling**을 통해  
클래스 불균형을 해소해  
준다.

&gt;&gt;

## STEP 2

몇 개의 그룹으로  
클러스터링 할지  
이어서 그래프를  
그려서 판단하여  
설정한다.

&gt;&gt;

## STEP 3

**GridSearchCV**를  
통해 모델의  
최적값을 찾고자  
하였다. 이렇게  
찾은 값을 하이퍼  
파라미터 값으로  
설정한다.

&gt;&gt;

## STEP 4

해당 하이퍼  
파라미터로  
**K-Means**를 사용한  
결과, 군집 2에서  
**CatBoostClassifier**가  
1번 클래스로 예측한  
비율이 30%, 군집  
4에서 클래스 0으로  
예측한 비율이 97%로  
**CatBoostClassifier**의  
예측과 유사하게  
군집이 형성되었다.

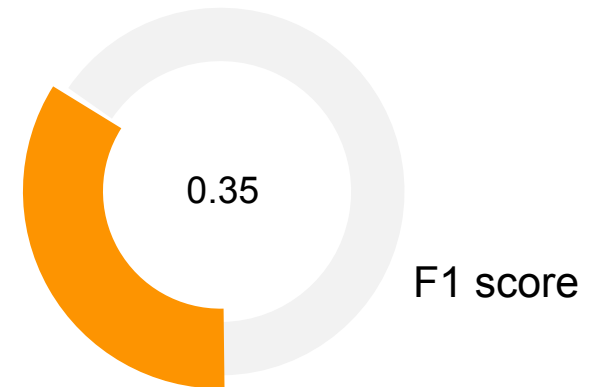
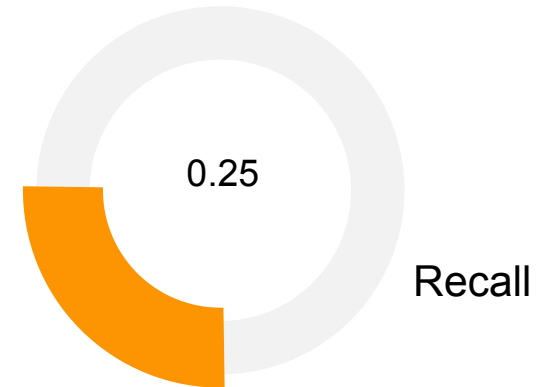
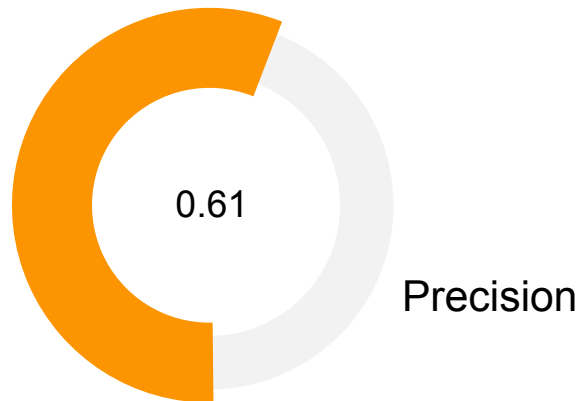
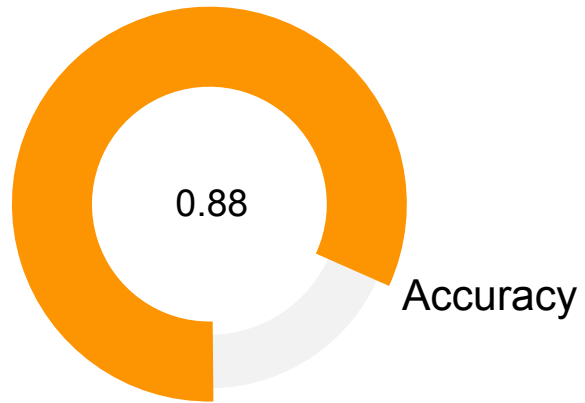
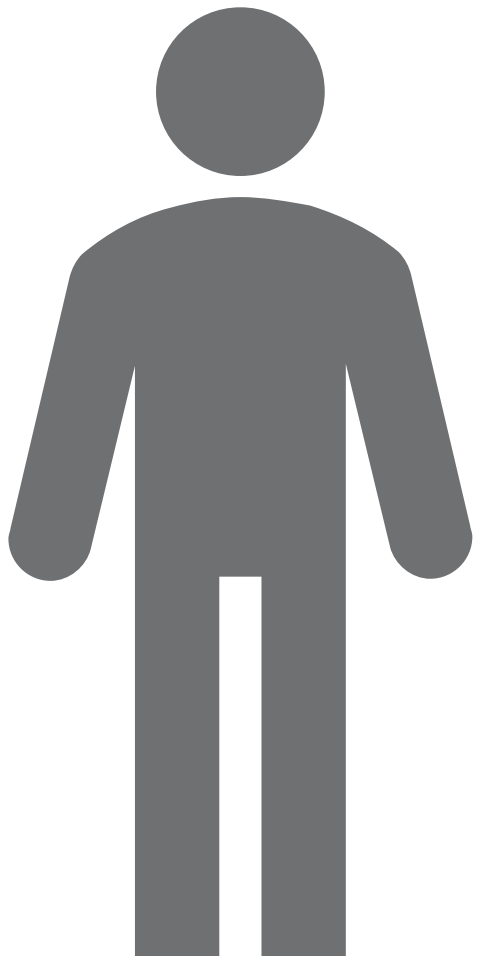
# Part 3.

최종 모델 예측 및  
타당성 검토



## Part 3, 최종 모델 예측

CatBoostClassifier





```
In [121]: X_val.groupby(['y_group']).y_pred.value_counts()
```

```
Out[121]: y_group  y_pred
0           0.0      555764
           1.0       97697
1           0.0      481660
           1.0       15502
2           0.0      473486
           1.0      207193
3           0.0      713291
           1.0       68882
4           0.0      452669
           1.0       13172
Name: y_pred, dtype: int64
```

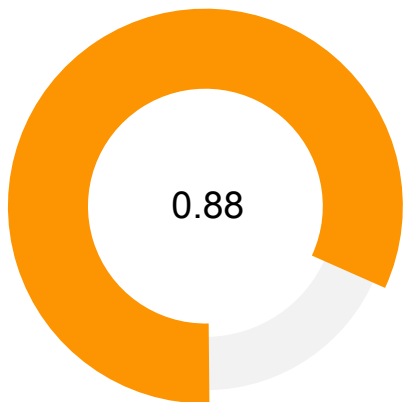
- KMeans 군집화 알고리즘으로 군집화한 결과

train\_set에서 1로 예측한 값이 2번 군집에서 1번

클래스의 비율로 30%가 나왔고, 4번 군집에서 0번

클래스의 비율이 97%로 나타나 유의미한 군집이

이루어졌음을 확인할 수 있다.



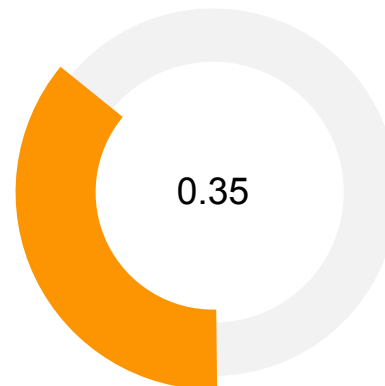
높은 Accuracy

전체 예측 중에 옳게 예측한 값을 의미하는 **Accuracy**의 값이 **88%**로 높게 형성되었다.  
이를 통해 모델의 예측이 높은 확률로 정확함을 확인할 수 있다.



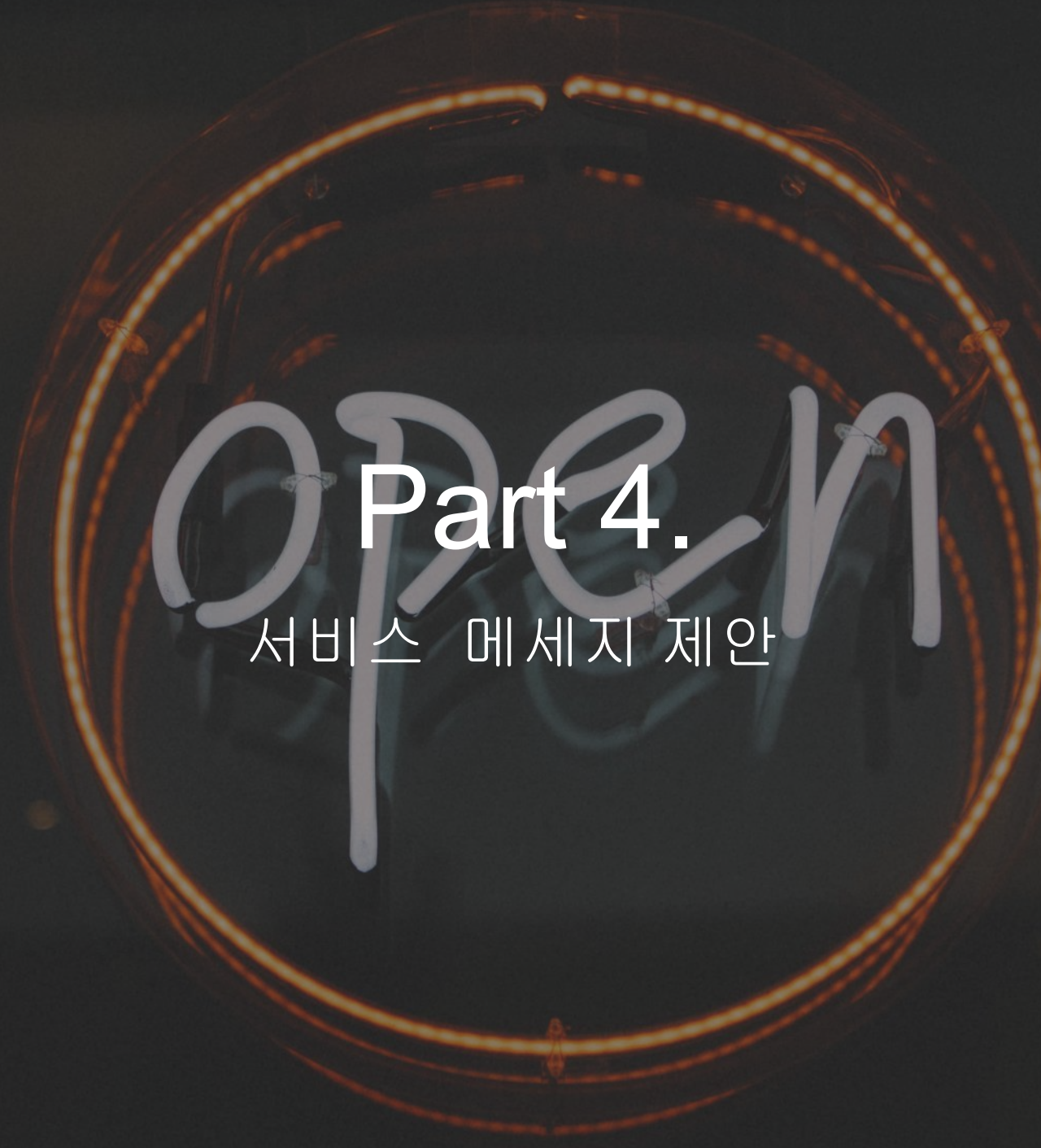
준수한 Precision

클래스를 1로 예측한 값 중에 실제 값이 1일 확률이 **0.61**로 클래스간 불균형이 매우 심한 데이터셋에서 학습한 것에 비해 준수하게 나타났다.



낮은 F1 score

다만 실제 클래스가 1인 것 중에 예측 클래스가 1임을 나타내는 **Recall**은 **0.25**로 낮은편인데, 그로 인해 **recall**과 **precision**의 균형을 나타내는 척도인 **F1 score**가 낮게 측정되었다.

A glowing orange circular light fixture, possibly a chandelier or a decorative lamp, is the central focus. It has a circular frame with multiple concentric rings of small, warm-toned lights. In the center of the circle, the word "OPEN" is written in large, white, stylized neon letters. The background is dark, and the overall atmosphere is warm and inviting.

Part 4.

서비스 메시지 제안



“Time”

“시간대별로 나눈 군집간의  
차이를 확인하였고 이에 각  
사용자 특성과 더불어  
시간대별로 다른 대출상품 제시”

# A

## Proposal

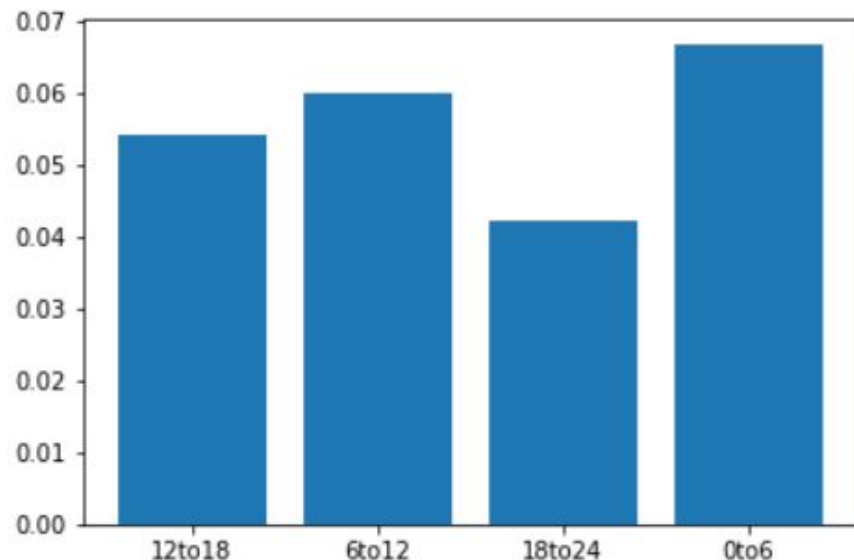
---

특히 **00시**에서 **06시**에 앱을 사용한 사람들이 실제 대출을 하는 비율이 높았고 **18시부터 24시까지**의 사용자들의 대출한도가 높다는 것을 확인하였다. 이를 통해 특정 시간대와 더불어 연령, 나이 등을 고려한 대출상품을 서비스 메시지로 제공하여 퍼포먼스 마케팅의 일환으로 사용한다.



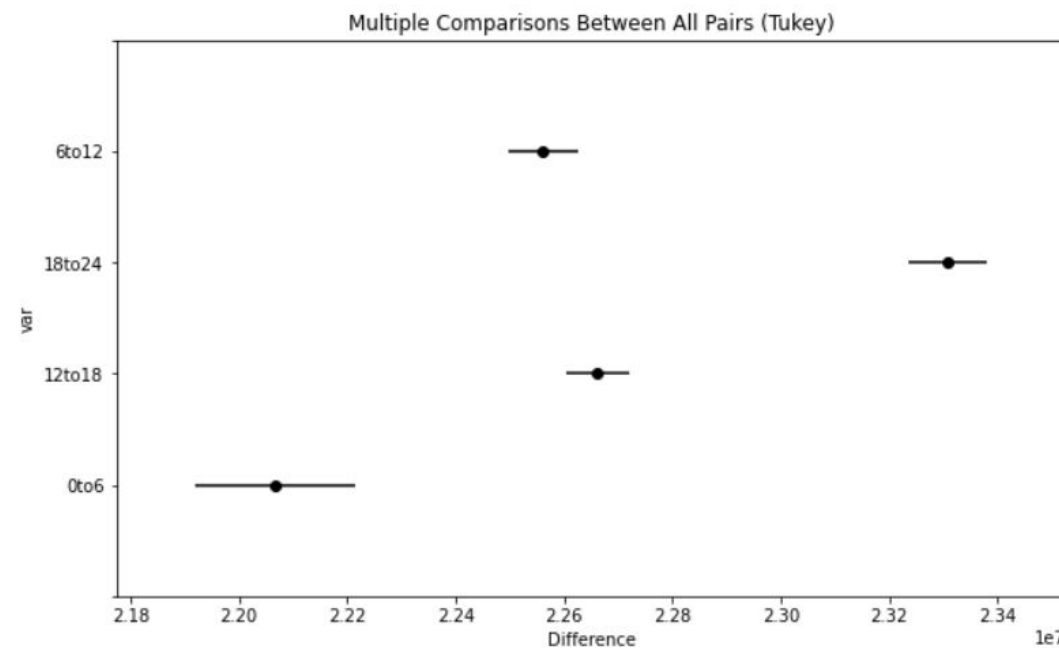


# B



## Specific

다중 비교를 위해 **Tukey** 검정을 실시하였고 결과적으로 18시부터 24시까지의 대출한도 값이 큰 것으로 나타났다. 하지만 실제 대출로 이어지는 비율이 적은 것으로 나타났다.



“limit-desired”

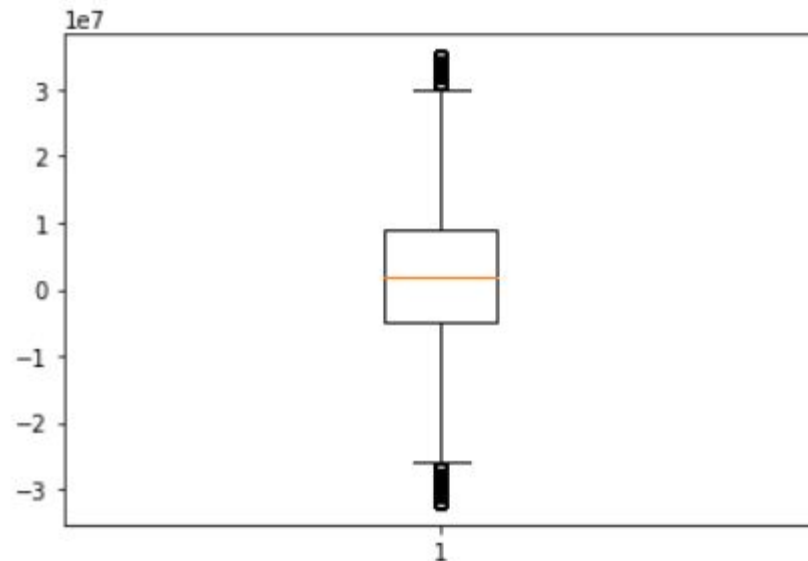
“대출을 신청한 사람들은  
대출한도가 본인이 희망하는  
금액보다 높을 경우 대출을 주로  
신청하였다.”

# A



## Specific

이상치 값을 일부 제거한 이후 **boxplot**과 통계치 요약값을 확인해 본 결과 대출 한도와 희망 금액이 차이가 많이 나지 않을 경우 주로 대출 신청을 많이 한 것을 확인할 수 있다.



```
count    5.058740e+05
mean      2.164701e+06
std       1.257382e+07
min       -3.200000e+07
25%       -5.000000e+06
50%        2.000000e+06
75%        9.000000e+06
max        3.500000e+07
Name: limit_desired, dtype: float64
```



# Q&A

감사합니다