신한은행 빅데이터 해커톤

데이런

김지수 박세라 송지우 윤서희 임아현

목차

개요

한경 금융

"야식 자주 먹으면 투자 성향 공격적?"...데이터 결합 싹 틔우는 신한금융

박진우 기자

입력 2020.12.02 17:42 수정 2020.12.03 01:41 지면 A14

- 유럽의 증권사에서 소비 성향을 투자상품 추천에 활용해 성공한 사례
  - 데이터 3법 시행을 계기로 본격적인 분석 작업을 시작하는 추세
- 현재 금융권에서 MZ세대를 고객으로 유치하기 위해 경쟁이 치열

개요

▪ 데이터 분석 목표

- 신한 카드 거래 고객 중 현재 증권사와 거래를 하고있는 고객의 특징 파악

  > 이를 활용하여 증권 성향의 고객을 예측하는 알고리즘 개발

- 증권 성향 고객에게 홍보나 혜택 등의 향후 실제 활용할 수 있는 방안을 고안

데이터 변수 설명

P1	P2	P3	P4	P5	P6	P7	B1	B2	B3	...	B165	B166	B167
M	20대_후	1	1	0	0	A은행	0	0	0	...	0	0	4350000
M	50대_후	1	0	0	0	A은행	0	0	0	...	0	0	580000
M	40대_후	1	1	0	0	A은행	0	0	0	...	0	0	1950000
F	60대_초	1	1	0	0	A은행	0	0	0	...	0	0	43000000
M	30대_후	0	1	0	0	B은행	0	0	0	...	0	0	4910000

C1	E1	E2	E3	E4	E5	E6
210	0	0	11111111110	0	111111111111	1100000000
40	0	0	0	0	111110111111	10000000000
50	0	1101101111	0	0	111101101111	101101101111
50	0	0	0	0	111111111111	111111111111
40	0	111111111111	0	0	111111111111	111111111111

신한 카드 결제 정보 데이터

- P3 : 은행활동고객TF
- P4 : 카드우수고객TF
- P5 : 금투활동고객TF
- P6 : 라이프활동고객TF
- B : 166개 업종 별 금액 + 총 결제 금액
- C : 결제 횟수
- E : 패턴코드 (12자리)

데이터 변수 설명

- **증권 성향 고객**

- 카드우수고객TF(P4), 금투활동고객TF(P5)가 모두 1인 고객
- 즉, 카드도 많이 사용하고 금융 투자도 많이 하는 고객을 증권 성향 고객

  > 신한투자증권을 활용하여 증권사에서 거래할 확률이 높을 것으로 판단

데이터 전처리

- Pandas와 Numpy 패키지를 설치하여 데이터 분석을 진행하였다.
- 데이터 시각화를 위해 matplotlib 패키지를 설치하였다.
- df = 473228 rows x 181 columns

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['font.family'] = 'Malgun Gothic'
matplotlib.rcParams['font.size'] = 15
matplotlib.rcParams['axes.unicode_minus'] = False
from google.colab import drive
drive.mount('/content/drive')

filename = '/content/drive/MyDrive/해커톤 준비/임아현/data_033/data_033.csv'
df = pd.read_csv(filename, encoding='cp949')
```

데이터 전처리

- 은행활동고객TF(P3), 카드우수고객TF(P4), 금투활동고객TF(P5), 라이프활동고객TF(P6) 컬럼이 모두 '0'인 고객은 신한은행 계좌가 없고, 신한카드를 주로 이용하지 않고, 신한투자증권을 이용하지 않고, 신한라이프에 미가입된 고객이다.
- 이러한 고객은 해당 사에서 증권을 구매할 가능성이 매우 낮은 고객들로 판단하여 분석에서 제외하였다.
- df = 406023 rows x 181 columns

```
filt = (df['P3']==0) & (df['P4']==0) & (df['P5']==0) & (df['P6']==0)
df = df[~filt]
df
```

P1	P2	P3	P4	P5	P6	P7	B1	B2	B3	...	B165	B166	B167	C1	E1	E2	E3	E4	E5	E6
M	20대_후	1	1	0	0	A은행	0	0	0	...	0	0	4350000	210	0	0	11111111110	0	111111111111	1100000000
M	50대_후	1	0	0	0	A은행	0	0	0	...	0	0	580000	40	0	0	0	0	111110111111	10000000000
M	40대_후	1	1	0	0	A은행	0	0	0	...	0	0	1950000	50	0	1101101111	0	0	111101101111	101101101111
F	60대_초	1	1	0	0	A은행	0	0	0	...	0	0	43000000	50	0	0	0	0	111111111111	111111111111
M	30대_후	0	1	0	0	B은행	0	0	0	...	0	0	4910000	40	0	111111111111	0	0	111111111111	111111111111

데이터 전처리

- 데이터를 전체적으로 훑어본 결과 E1~E6까지 패턴코드였고, E4는 론이용패턴코드임을 확인하였다.
- 신한카드 이용 고객 중 현재 증권사와 거래를 하고 있는 고객의 특징을 정의하기 위해 카드우수고객TF(P4), 금투활동고객TF(P5) 데이터를 집중해서 분석을 진행하였다.
- 이 과정에서 론이용패턴코드(E4)는 증권 성향이 높은 고객을 설명해주기에 모호한 부분이 있다고 판단하여 E4를 분석에서 제외하였다.
- df = 406023 rows x 180 columns [최종적으로 분석에 사용하게 된 데이터]

```
df = df.drop(columns=['E4'])
df
```

P1	P2	P3	P4	P5	P6	P7	B1	B2	B3	...	B164	B165	B166	B167	C1	E1	E2	E3	E5	E6
M	20대_후	1	1	0	0	A은행	0	0	0	...	0	0	0	4350000	210	0	0	11111111110	111111111111	1100000000
M	50대_후	1	0	0	0	A은행	0	0	0	...	0	0	0	580000	40	0	0	0	111110111111	10000000000
M	40대_후	1	1	0	0	A은행	0	0	0	...	19000	0	0	1950000	50	0	1101101111	0	111101101111	101101101111
F	60대_초	1	1	0	0	A은행	0	0	0	...	0	0	0	43000000	50	0	0	0	111111111111	111111111111
M	30대_후	0	1	0	0	B은행	0	0	0	...	0	0	0	4910000	40	0	111111111111	0	111111111111	111111111111

데이터 분석

- 데이터 간의 관계를 파악하고 새로운 출력 데이터를 만들어낸다.

1. 금융투자 유무에 따른 '연령대별 고객 분포'는 어떠한가?
2. 금융투자 유무에 따른 '연령대별 성별 비중'은 어떠한가?
3. 금융투자 유무에 따른 '업종별 차이'는 어떠한가?

데이터 분석

```
a = (df['P5'] == 1)
df_T = df[a]
df_T
```

금융투자 유

df_T = 47691 rows x 180 columns

	P1	P2	P3	P4	P5	P6	P7	B1	B2	B3	...	B164	B165	B166	B167	C1	E1	E2	E3	E5	E6
8	M	30대_후	0	0	1	0	B은행	0	0	0	...	27000	0	0	1130000	40	0	0	0	111110000000	111100000000
27	M	40대_후	1	1	1	1	A은행	0	0	0	...	0	0	0	2150000	130	0	100000	0	100010100101	100000100000
29	F	50대_초	1	1	1	1	A은행	0	0	0	...	0	0	0	3650000	100	0	10010	0	111111111111	111111111000
33	F	30대_후	1	1	1	0	A은행	0	0	0	...	0	0	0	3730000	70	111111111111	100000000000	0	111111111111	111111111111
38	F	40대_후	1	1	1	0	A은행	0	0	0	...	0	0	0	7620000	160	0	0	0	111111111111	111111111111

```
b = (df['P5'] == 0)
df_F = df[b]
df_F
```

금융투자 무

df_F = 358332 rows x 180 columns

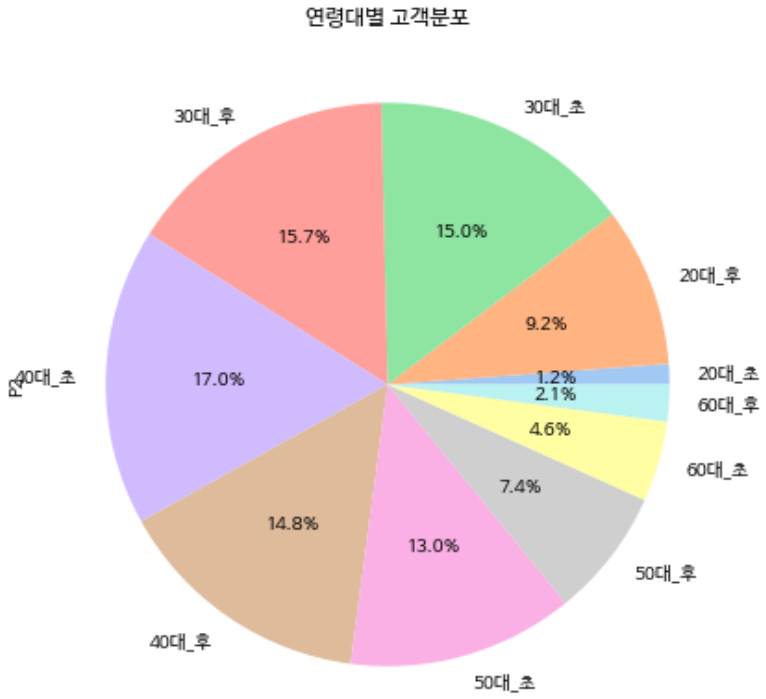
	P1	P2	P3	P4	P5	P6	P7	B1	B2	B3	...	B164	B165	B166	B167	C1	E1	E2	E3	E5	E6
0	M	20대_후	1	1	0	0	A은행	0	0	0	...	0	0	0	4350000	210	0	0	11111111110	111111111111	1100000000
1	M	50대_후	1	0	0	0	A은행	0	0	0	...	0	0	0	580000	40	0	0	0	111110111111	10000000000
2	M	40대_후	1	1	0	0	A은행	0	0	0	...	19000	0	0	1950000	50	0	1101101111	0	111101101111	101101101111
3	F	60대_초	1	1	0	0	A은행	0	0	0	...	0	0	0	43000000	50	0	0	0	111111111111	111111111111
4	M	30대_후	0	1	0	0	B은행	0	0	0	...	0	0	0	4910000	40	0	111111111111	0	111111111111	111111111111

데이터 분석

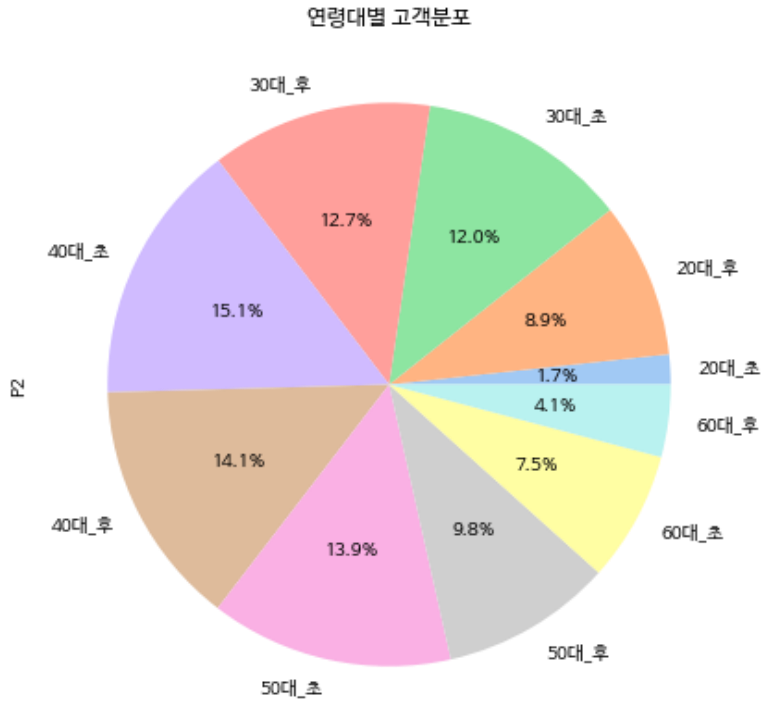
금융투자 유무에 따른 '연령대별 고객 분포'

```
age=df_T['P2'].value_counts().sort_index()
plt.title("연령대별 고객분포")
age.plot(kind = 'pie', figsize=(7,7),autopct='%1.1f%%',colors = color)
plt.show()
```

금융투자 유



금융투자 무

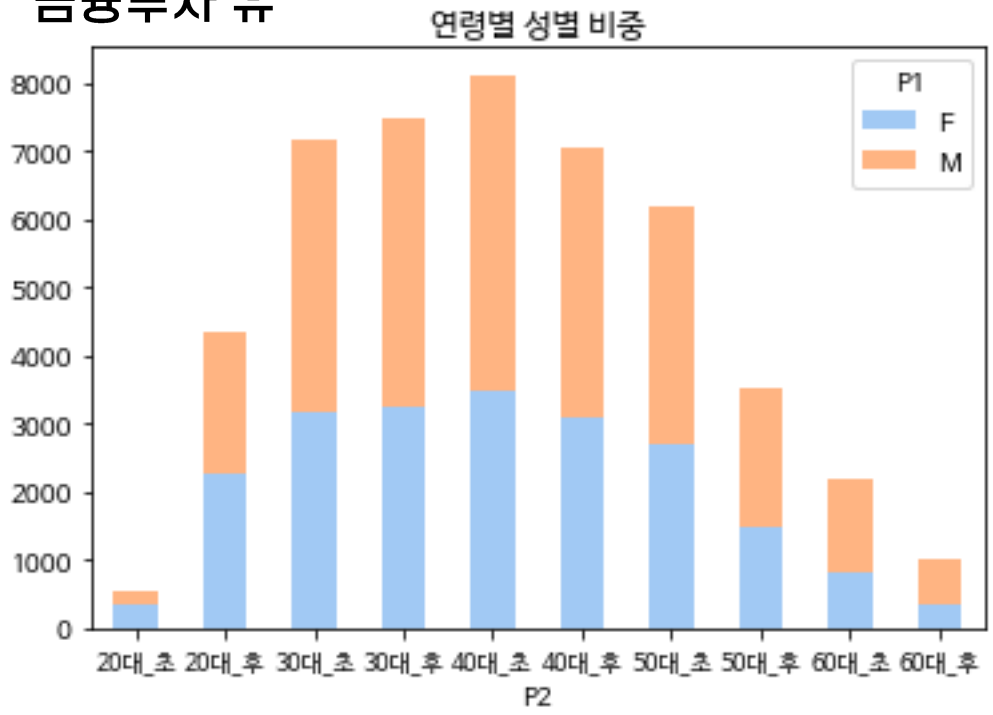


데이터 분석

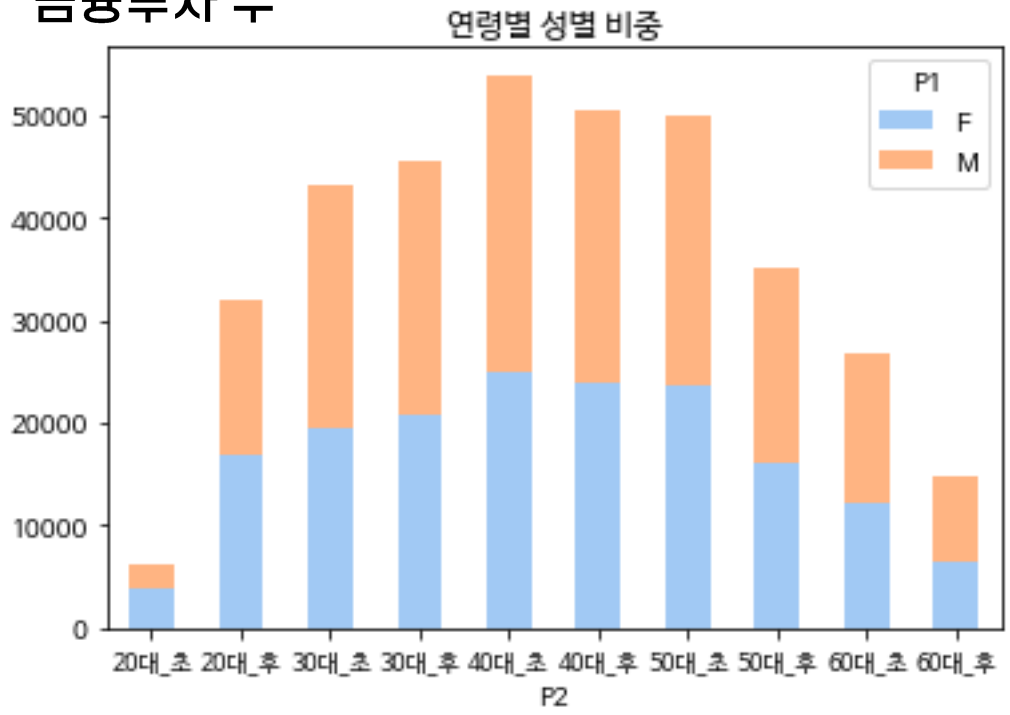
금융투자 유무에 따른 '연령대별 성별 비중'

```
stacked_bar_df_T = df_T.groupby(["P2","P1"]).size().unstack()
stacked_bar_df_T.plot(kind='bar', stacked=True, color=color)
plt.title("연령별 성별 비중")
plt.xticks(rotation=0, fontsize=9)
plt.show()
```

금융투자 유



금융투자 무



데이터 분석

금융투자 유무에 따른 '업종별 차이'

숙박	B1-B6	교통	B7-B11
쇼핑	B13-B21, B32, B77-B78	농수산	B22-B29
식료품	B30-B31, B35-B42	가구	B43-B54
사무/전자기기	B55-B60	의류/미용	B63-B76, B152-B153
취미	B79-B96	스포츠/문화/레저	B97-B105, B121-B122, B137-B138
보험/의료	B106-B107, B139-B149	교육	B108, B155-B156
자동차	B118-B120, B132, B157-B166	가정생활/서비스	B135-B136, B154
업종	B124-B127, B151-B152	서비스	B129-B132

- 각 업종별로 금융 투자 유무에 따라 어떤 부분에 많은 금액을 소비하는지 조사하고자, 기존 데이터에서 비슷한 분야를 업종별로 묶어서 분석을 진행하였다.
- 일상에서 자주 사용하는 곳이 아니고, '1개월'이라는 주어진 기간 안에 드문 확률로 사용되는 소비 데이터는 분석에서 제외하였다. EX) 예식장, 장의사

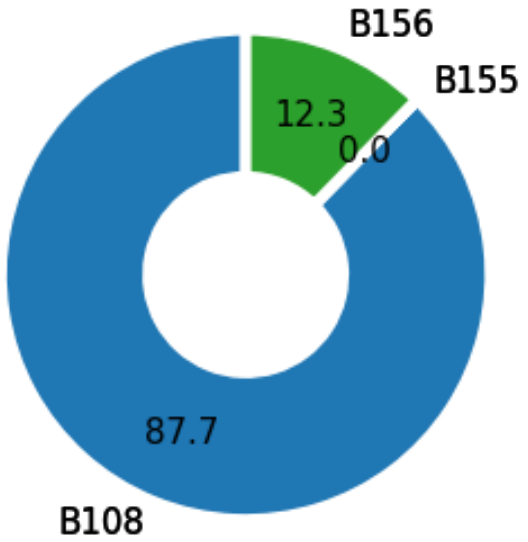
데이터 분석

금융투자 유무에 따른 '업종별 차이'가 작은 업종 [교육]

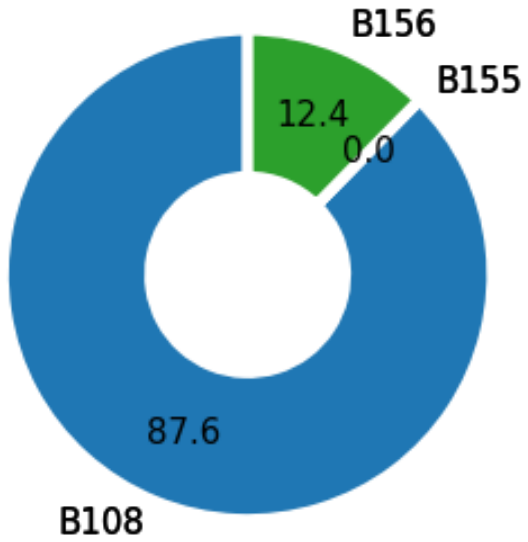
```
df_T_edu = df_T.loc[:, ['B108','B155','B156']]
df_T_edu_s = df_T_edu[['B108','B155','B156']].sum()

values = df_T_edu_s
labels = df_T_edu.columns
wedgeprops = {'width':0.6, 'edgecolor':'w', 'linewidth':5}
plt.figure(figsize=(5, 5))
plt.pie(values, labels=labels, autopct='%.1f', pctdistance=0.7, wedgeprops=wedgeprops, startangle=90)
plt.show()
```

금융투자 유



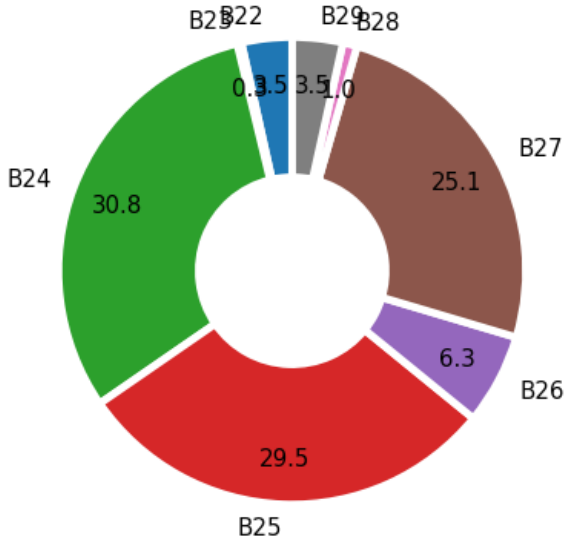
금융투자 무



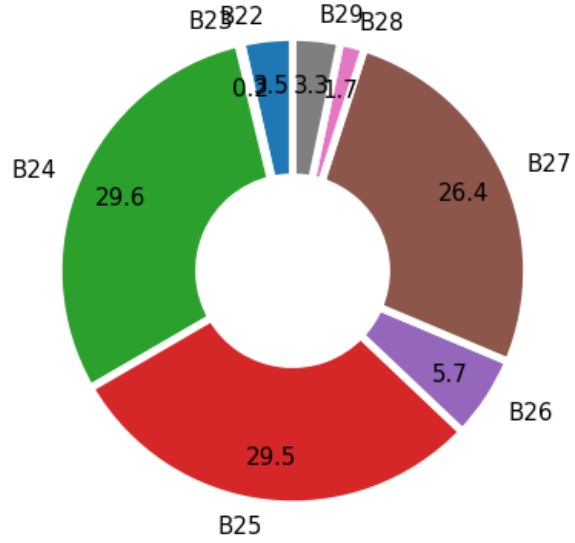
데이터 분석

금융투자 유무에 따른 '업종별 차이'가 작은 업종 [농수산, 식료품]

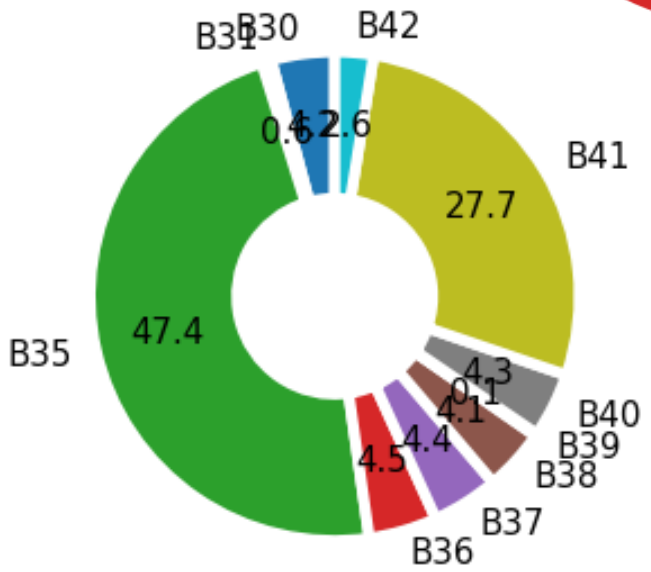
금융투자 유



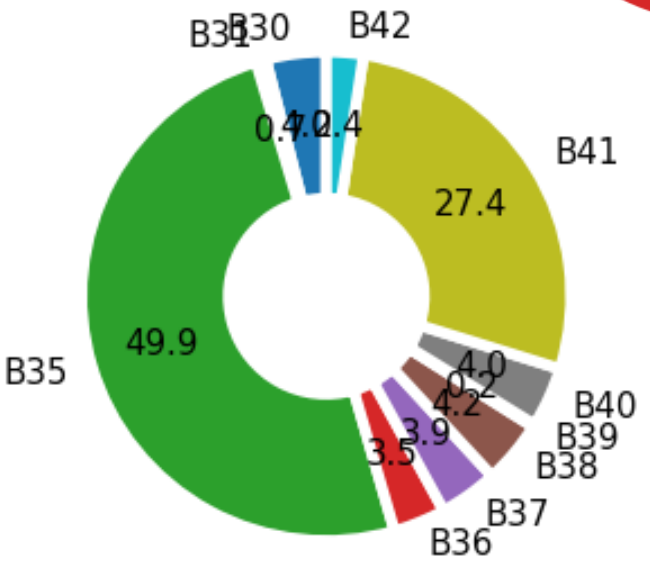
금융투자 무



금융투자 유



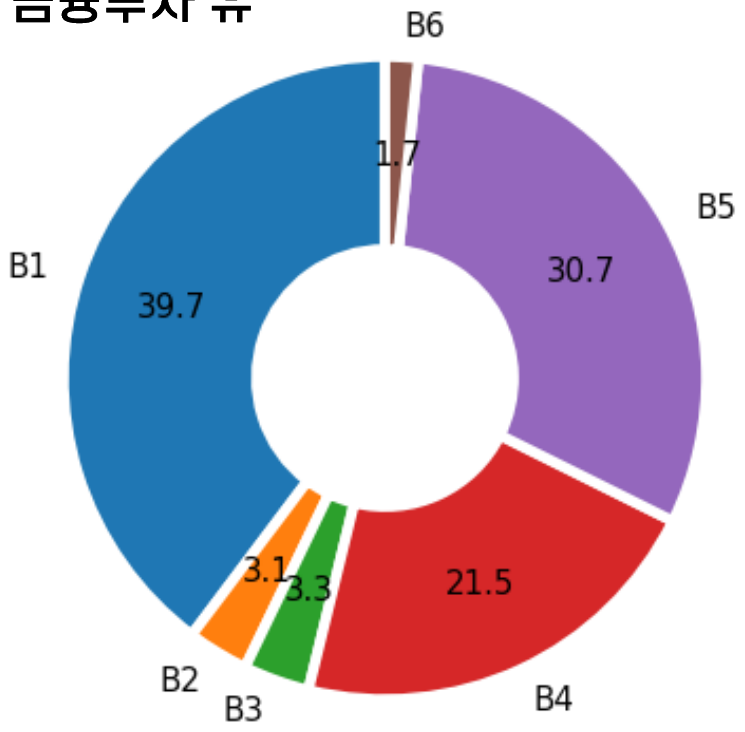
금융투자 무



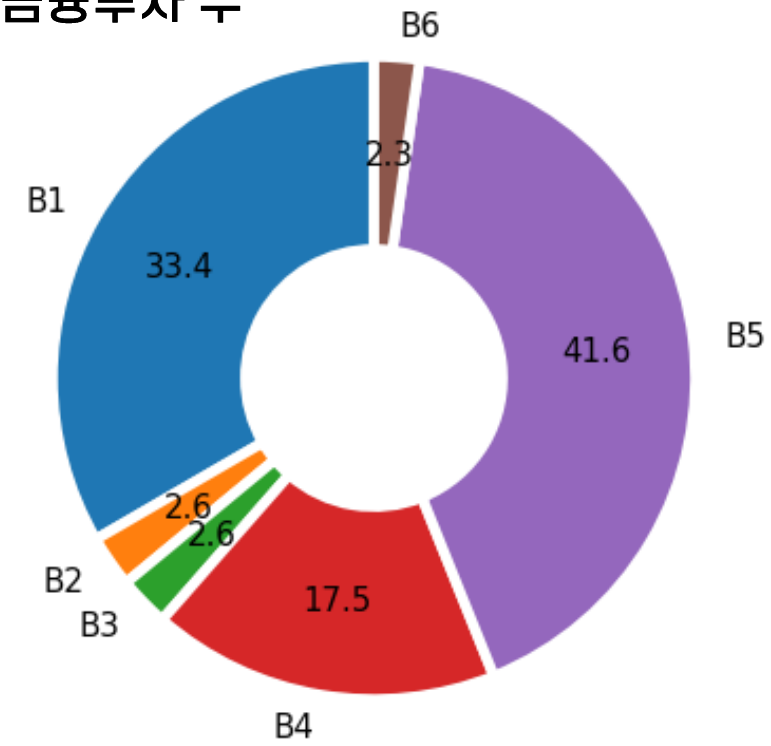
데이터 분석

금융투자 유무에 따른 '업종별 차이'가 큰 업종 [숙박]

금융투자 유



금융투자 무

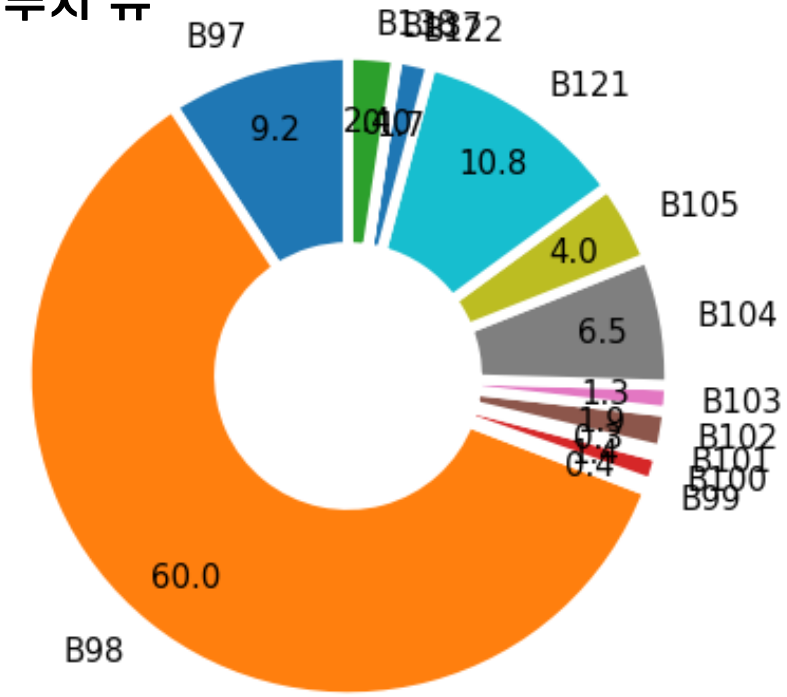


- 금융 투자를 하는 고객들이 특급호텔(B1)에서 숙박하는 비중이 6.3% 높다.

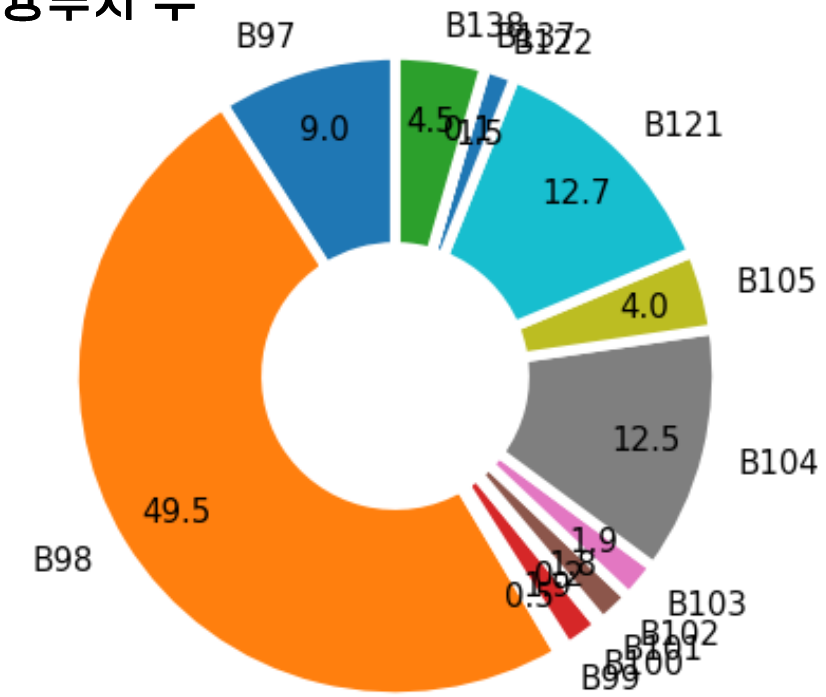
데이터 분석

금융투자 유무에 따른 '업종별 차이'가 큰 업종 [스포츠/문화/레저]

금융투자 유



금융투자 무

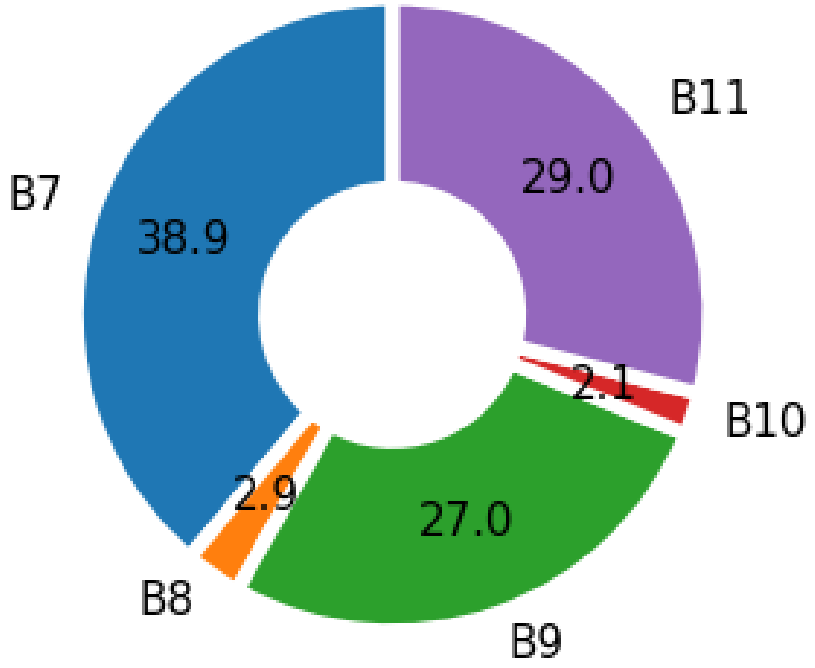


- 금융 투자를 하는 고객들이 실외 골프장(B98)을 이용하는 비중이 10.5% 높다.

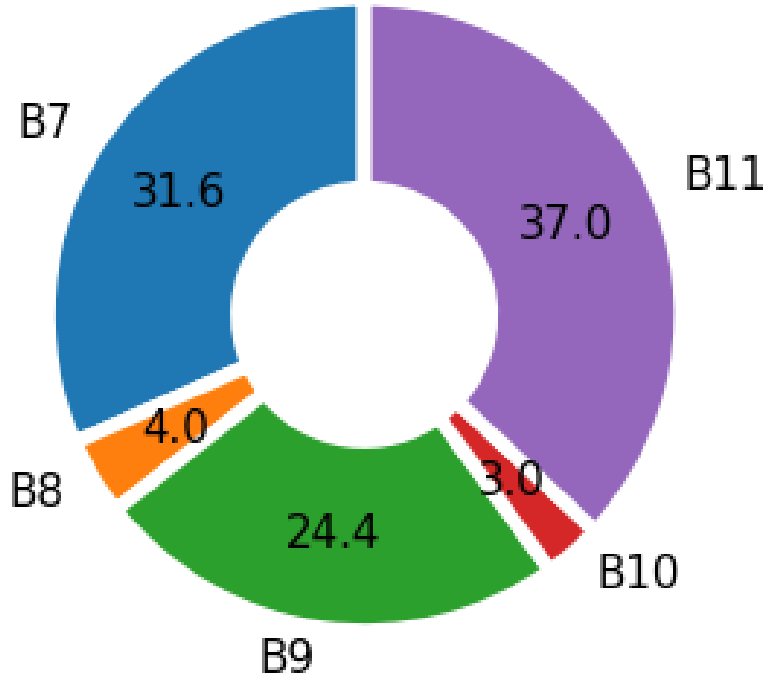
데이터 분석

금융투자 유무에 따른 '업종별 차이'가 큰 업종 [교통]

금융투자 유



금융투자 무



- 금융 투자를 하는 고객들이 항공사(B7)를 이용하는 비중이 7.3% 높다.

모델링

XGBOOST

1. 모듈 Import

```
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

2. 데이터 전처리 > 금융투자TF(P5)와 B1-B167만 남긴다.

```
df = df.drop(columns=['P1', 'P2', 'P3', 'P4', 'P6', 'P7', 'C1', 'E1', 'E2', 'E3', 'E4', 'E5', 'E6'])
df
```

	P5	B1	B2	B3	B4	B5	B6	B7	B8	B9	...	B158	B159	B160	B161	B162	B163	B164	B165	B166	B167
0	0	0	0	0	0	190000	0	101000	0	54000	...	0	0	0	70000	76000	0	0	0	0	4350000
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	580000
2	0	0	0	0	0	0	0	0	1000	0	...	0	0	0	0	110000	0	19000	0	0	1950000
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	43000000
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	20000	0	0	0	0	0	4910000

모델링

XGBOOST

3. 데이터 가공하기 > P5(금투TF) : 목표변수, B1-B167 : 설명변수

```
X = df.loc[:, 'B1':'B167'] #설명변수
Y = df.loc[:, 'P5'] #목표변수
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.3, random_state = 7)
X_train
```

	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	...	B158	B159	B160	B161	B162	B163	B164	B165	B166	B167
36540	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	260000
55370	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	900000
347912	0	0	0	0	0	0	0	0	0	0	...	0	0	0	93000	0	0	0	0	0	760000
269198	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	820000
113748	0	0	0	0	0	0	0	0	0	0	...	0	0	0	2112000	0	0	0	0	0	4010000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
235075	0	0	0	0	0	0	0	0	306000	0	...	0	0	0	0	0	0	0	0	0	2190000
10742	0	0	0	0	0	0	0	0	0	0	...	0	0	0	30000	0	0	0	0	0	580000
49689	0	0	0	0	0	0	0	0	17000	0	...	0	0	0	0	0	0	0	0	0	840000

모델링

XGBOOST

4. 모델 훈련시키기

```
model = XGBClassifier()
model.fit(X_train, y_train)
print(model)
```

5. 예측 수행하기

```
Y_pred = model.predict(X_test)
predictions = [round(value) for value in Y_pred]
```

6. 모델 평가하기

```
accuracy = accuracy_score(Y_test, predictions)
print("Accuracy: %.2f%%" % (accuracy*100.0))
```

```
Accuracy: 89.89%
```

Accuracy : 89.89%

모델링

CATBOOST

1. 데이터 전처리

```
X = df.loc[:, 'B1':'B167'] #설명변수
Y = df.loc[:, 'P5'] #목표변수

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.3, random_state = 7)
X_train
```

	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	...	B158	B159	B160	B161	B162	B163	B164	B165	B166	B167
36540	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	260000
55370	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	900000
347912	0	0	0	0	0	0	0	0	0	0	...	0	0	0	93000	0	0	0	0	0	760000
269198	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	820000
113748	0	0	0	0	0	0	0	0	0	0	...	0	0	0	2112000	0	0	0	0	0	4010000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
235075	0	0	0	0	0	0	0	0	306000	0	...	0	0	0	0	0	0	0	0	0	2190000

모델링

CATBOOST

2. 모듈 Import

```
import numpy as np
from catboost import CatBoostClassifier, Pool

from sklearn.ensemble import RandomForestClassifier
from lightgbm import LGBMClassifier
from catboost import CatBoostClassifier
from sklearn.linear_model import LogisticRegression
```

3. 데이터 가공하기

```
print('train shape: ', X_train.shape)
print('test shape: ', X_test.shape)
```

```
train shape:  (331259, 167)
test shape:  (141969, 167)
```

```
print('train shape: ', Y_train.shape)
print('test shape: ', Y_test.shape)
```

```
train shape:  (331259,)
test shape:  (141969,)
```

모델링

CATBOOST

4. 모델 훈련시키기

```
model = CatBoostClassifier()
model.fit(X_train, Y_train)
print(model)
```

```
Learning rate set to 0.122757
0:      learn: 0.5947707        total: 200ms    remaining: 3m 19s
1:      learn: 0.5228800        total: 373ms    remaining: 3m 5s
2:      learn: 0.4690816        total: 590ms    remaining: 3m 16s
3:      learn: 0.4313746        total: 774ms    remaining: 3m 12s
4:      learn: 0.4030955        total: 982ms    remaining: 3m 15s
5:      learn: 0.3821298        total: 1.2s     remaining: 3m 18s
```

5. 예측 수행하기

```
Y_pred = model.predict(X_test)
predictions = [round(value) for value in Y_pred]

accuracy = accuracy_score(Y_test, predictions)
print("Accuracy: %.2f%%" % (accuracy*100.0))
```

```
Accuracy: 89.87%
```

Accuracy : 89.87%

평가 및 개선사항

- 단기 데이터로 분석을 진행하였기 때문에 모델링의 결과가 정확하지 않을 수 있다.
  - 장기 데이터로 분석하는 경우에 활용도가 높아지기 때문에, 누적된 데이터를 사용하는 것이 바람직하다고 판단된다.

- 금융 투자를 하는 사람들이 금융 투자를 하지 않는 사람들보다 높은 비중을 보이는 업종은 실외 골프장, 특급 호텔, 항공사이다.
  - 세 개의 업종을 중심으로 홍보와 활용 방안을 제시하고자 한다.

평가 및 개선사항

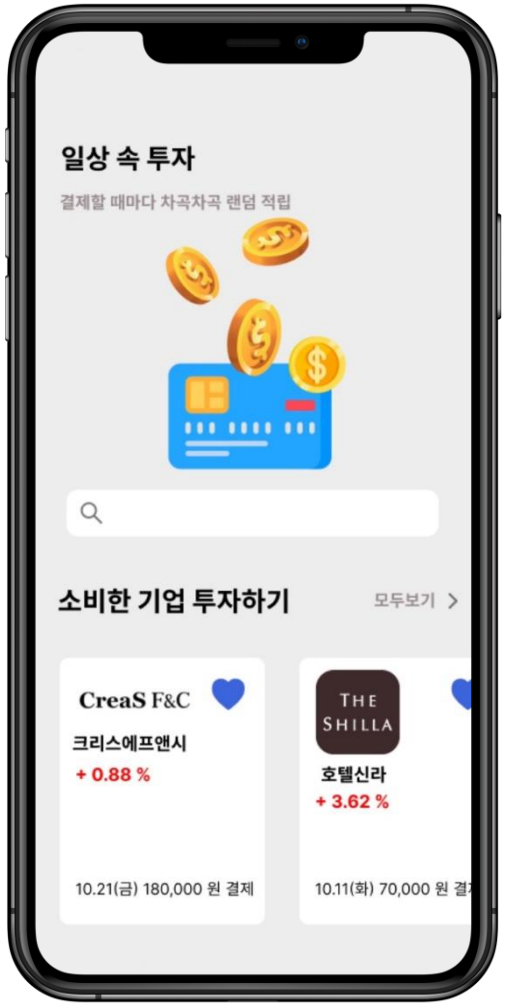
- MTS (Mobile Trading System)
  - 최근 개인 투자자 참여 확대 > MTS 이용 비중 증가
  - MZ 세대 유입 > 편의성을 위해 하나의 통합 앱으로 운영하는 추세

신한투자증권

<신한알파>

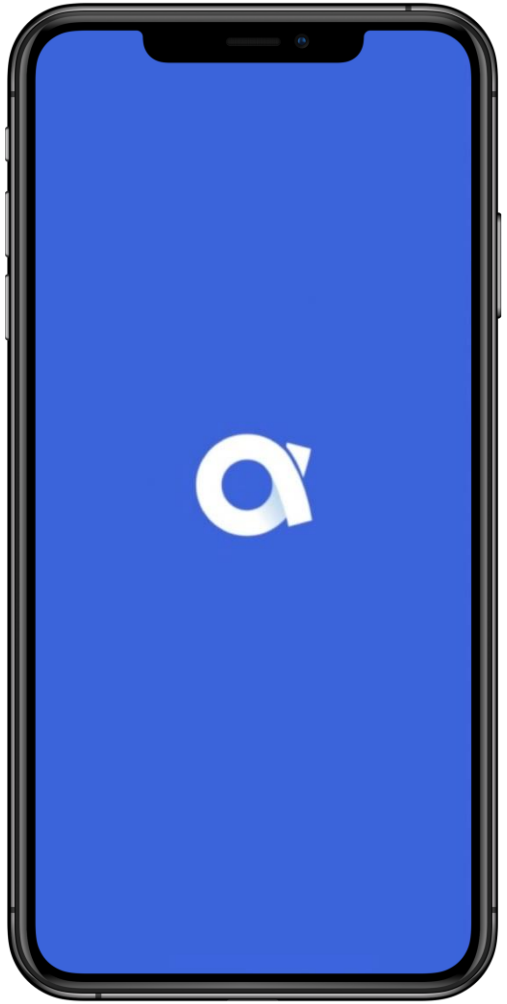
- 단순하고 깔끔한 기능으로 호평
- ‘앱인앱 형식’으로 추가 앱 설치 없이 100개 이상의 금융 서비스 제공
- 주식 잔고 ‘영수증‘ 서비스

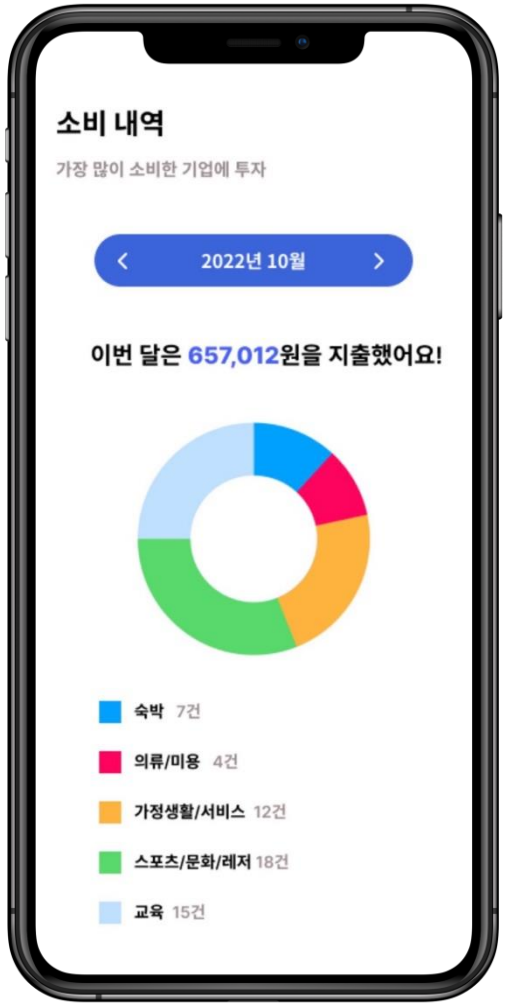
평가 및 개선사항

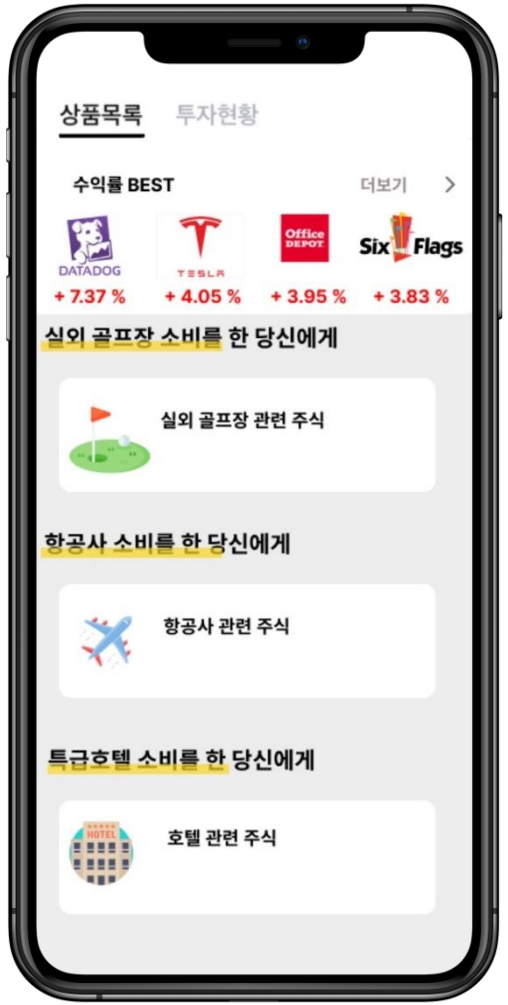


- 소액 자동 투자 시스템
  - 실외 골프장, 항공사, 특급호텔
  - 신한투자증권에서 사용 가능한 포인트 지급
  - 일정 포인트 이상 시, 자동으로 투자해주는 시스템
  - 주식 계좌가 없다면, 시스템 홍보 문자 전송

평가 및 개선사항




소비 내역
가장 많이 소비한 기업에 투자
2022년 10월
이번 달은 657,012원을 지출했어요!
숙박 7건
의류/미용 4건
가정생활/서비스 12건
스포츠/문화/레저 18건
교육 15건


상품목록
투자현황
수익률 BEST
더보기
DATADOG
+ 7.37 %
TESLA
+ 4.05 %
Office DEPOT
+ 3.95 %
Six Flags
+ 3.83 %
실외 골프장 소비를 한 당신에게
실외 골프장 관련 주식
항공사 소비를 한 당신에게
항공사 관련 주식
특급호텔 소비를 한 당신에게
호텔 관련 주식

평가 및 개선사항

1. MZ세대의 경우 투자 자체를 소액으로 하는 경우가 많다.
2. 경제 지식이 부족하고 공부하지 않은 상태로 게임처럼 투자한다.

전문가와 AI가 판단한 종목에 투자하는 것이 수익률이 높을 확률이 크다.

감사합니다