UCI Machine Learning Repository

Spain

고객 세그멘테이션 분석

홍혜린

분석 순서

※ Spain의 군집 목적

- Spain CRM 생성.
- 소비 금액에 따른 회원 분류 후 군집 진행
- 1차 군집화
- 군집화 정도를 보고 가공, 2차 군집화

※ 심화

- Spain CRM을 통해 그들의 소비 패턴을 파악
- 소비패턴 :
  - 특정 구간 구매자
  - 구매 상품에 대한 분석 :
    · split을 통한 상품 특징 추출

1. 데이터

- UCI machine learning repository에서 Retail 데이터 사용.

UCI Dataset URL (https://archive.ics.uci.edu/ml/datasets/Online+Retail+II)

- 국가 별로 retail 데이터 중 스페인을 선택해 스페인 retail data 분석을 진행.

- Regression 으로 dataset을 분석, 군집화를 진행하였습니다.

2. 전처리

0이 아닌 값들 제외

```
56]: retail_df = retail_df[retail_df['Quantity'] > 0]
     retail_df = retail_df[retail_df['UnitPrice'] > 0]
     retail_df = retail_df[retail_df['CustomerID'].notnull()]
     print(retail_df.shape)
     retail_df.isnull().sum()
```

```
(2484, 8)
```

```
56]: InvoiceNo      0
     StockCode      0
     Description    0
     Quantity       0
     InvoiceDate    0
     UnitPrice      0
     CustomerID     0
     Country        0
     dtype: int64
```

고객 당 얼마 구매했는지 누적 = Total_amount

```
In [16]: t_amount = result['Quantity'] * result['UnitPrice']
         result['Total_amount'] = t_amount
         result
```

Out[16]:

	CustomerID	InvoiceNo	StockCode	Quantity	InvoiceDate	UnitPrice	Total_amount
0	12557.0	536944	22383	70	2010-12-03 12:20:00	1.65	115.50
1	12557.0	536944	22384	100	2010-12-03 12:20:00	1.45	145.00
2	12557.0	536944	20727	60	2010-12-03 12:20:00	1.65	99.00
3	12557.0	536944	20725	70	2010-12-03 12:20:00	1.65	115.50
4	12557.0	536944	20728	100	2010-12-03 12:20:00	1.45	145.00
...	...	...	...	...	...	...	...
144	12442.0	580955	21974	12	2011-12-06 14:22:00	1.45	17.40
145	12442.0	580955	23597	6	2011-12-06 14:22:00	2.95	17.70
146	12442.0	580955	22090	6	2011-12-06 14:22:00	2.95	17.70
147	12442.0	580955	21209	12	2011-12-06 14:22:00	0.39	4.68
148	12442.0	580955	21981	24	2011-12-06 14:22:00	0.39	9.36

149 rows × 7 columns

분석 들어가기 전

적은 고객으로 인한 분포 확인이 쉽지 않았습니다.

고객 31명

```
In [54]: retail_df['CustomerID'].unique()

Out[54]: array([12557., 17097., 12540., 12551., 12503., 12484., 12539., 12510.,
                12421., 12502., 12462., 12507., 12541., 12547., 12597., 12545.,
                12596., 12354., 12417., 12455., 12450., 12548., 12556., 12550.,
                12546., 12454., 12448., 12544., 12538., 12445., 12442.])
```

** 고객 별 소비 금액 등급 나눔 (target)

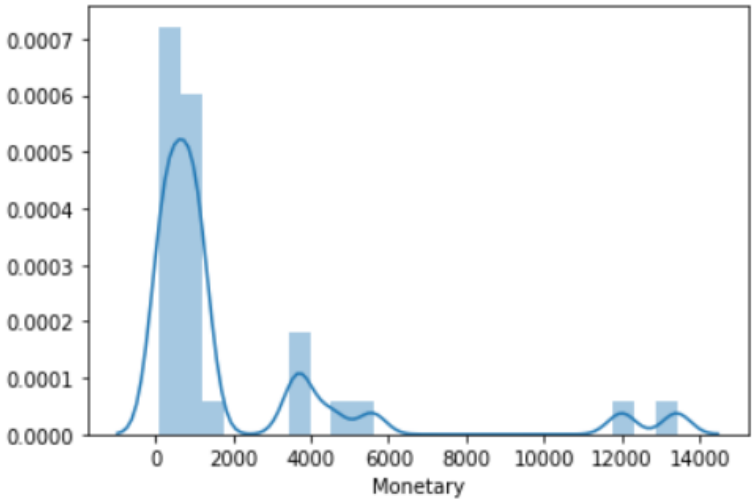
Silver, Gold, Diamond 회원으로 나눔

값이 편중되어 있어 로그를 써서 진행했다.

고객 별 소비별 금액에 따른 등급 나눔

```
sns.distplot(cust_df['Total_amount'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x282746d66c8>
```

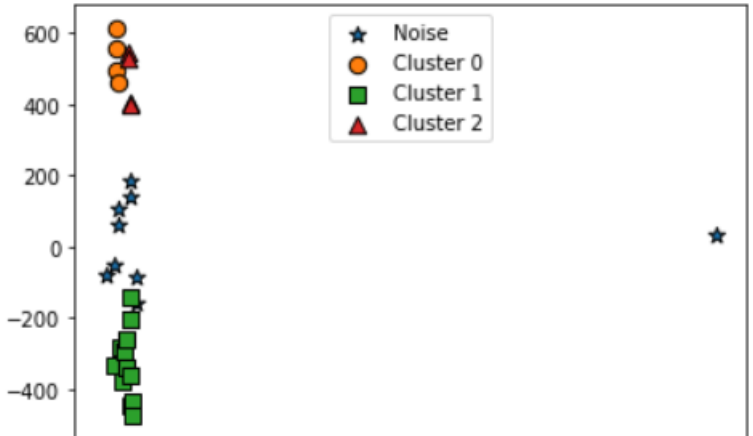


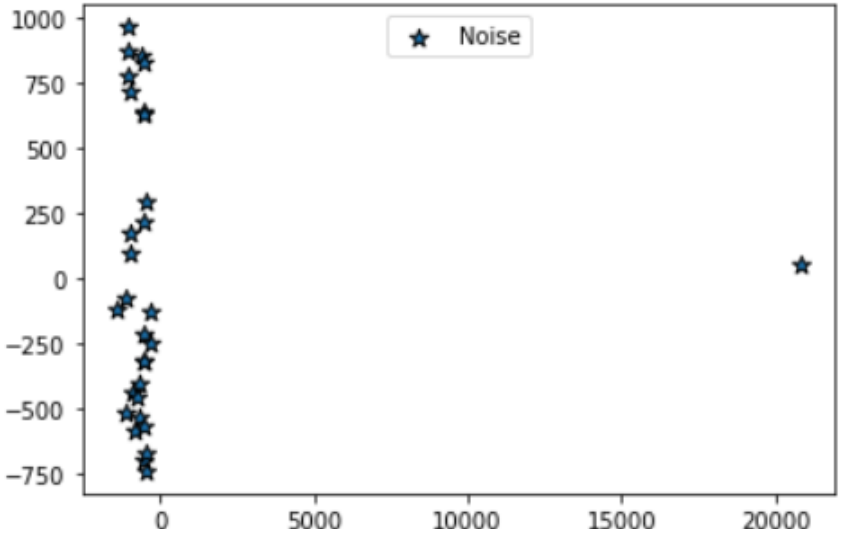
3. 1차 - DBSCAN 에 따른 군집화, 파라미터 설정에 따라 노이즈 확인

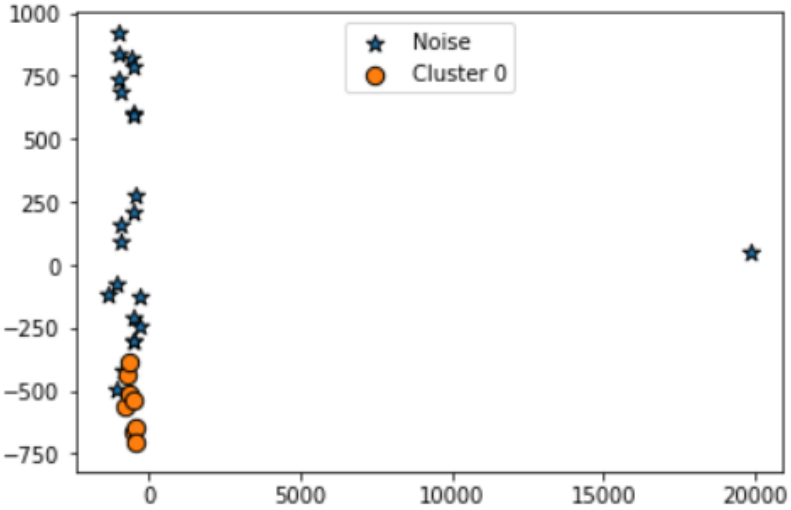
35]:

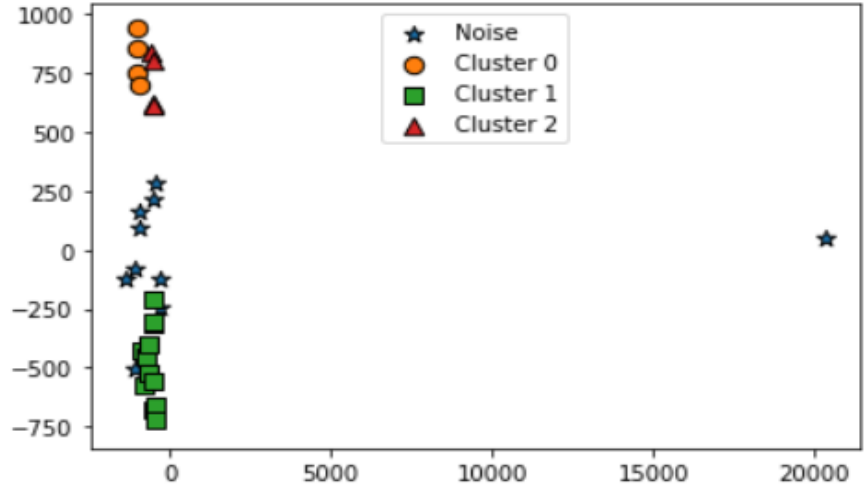
```
# 2차원으로 시각화하기 위해 PCA n_componets=2로 피처 데이터 세트 변환
pca = PCA(n_components=2, random_state=0)
pca_transformed = pca.fit_transform(cus_df2)
# visualize_cluster_2d( ) 함수는 ftr1, ftr2 컬럼을 좌표에 표현하므로 PCA 변환값을 해당 컬럼으로 생성
cus_df2['ftr1'] = pca_transformed[:,0]
cus_df2['ftr2'] = pca_transformed[:,1]

visualize_cluster_plot(dbscan, cus_df2, 'dbscan_cluster', iscenter=False)
#eps = 150, min_samples = 4
```









파라미터 설정에 따라 노이즈 확인하고
적절히 군집이 될 때까지 반복한다.

5. 실 데이터와 확인하여 노이즈 여부 결정

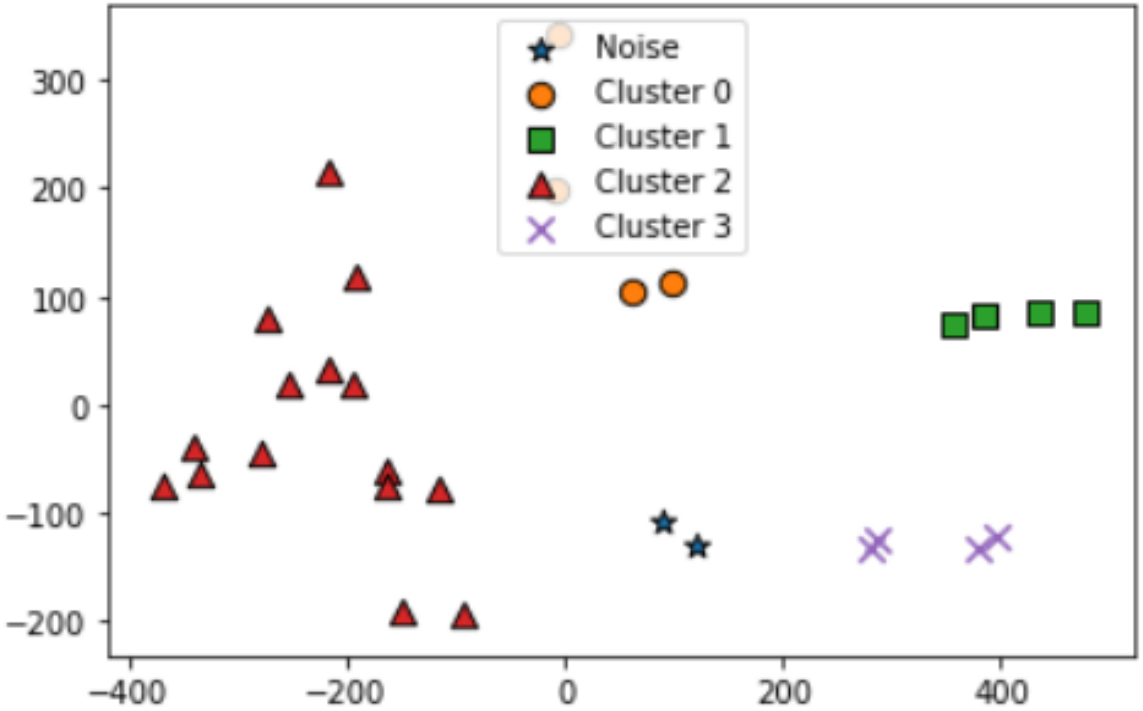
DBSCAN을 통한 군집화를 진행했는데,

노이즈로 인해 시각화하는데 어려움이 있었다.

노이즈 제거 후에 다시 DBSCAN 진행한다.

2차 - DBSCAN 에 따른 군집화

→ 명확한 군집을 형성함



6. 각 군집마다 어떤 고객 군인지 확인

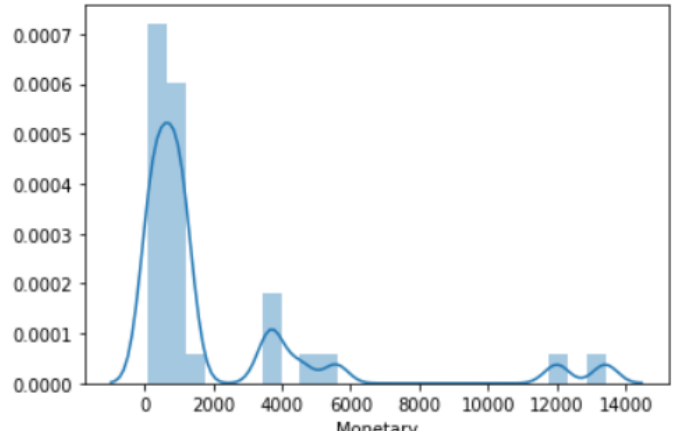
★기존에 소비누적 그래프에서 분산된 부분 = 다이아몬드 회원

앞서 노이즈는 다이아몬드 회원이라고 추정 → 적중 48.2%

고객 별 소비별 금액에 따른 등급 나눔

```
sns.distplot(cust_df['Total_amount'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x282746d66c8>
```




```
dfff[dfff['Grade']==dfff['dbscan_cluster']].count()
#11개
```

```
CustomerID        11
InvoiceDate       11
InvoiceNo         11
Quantity          11
Total_amount      11
Grade             11
dbscan_cluster    11
ftr1              11
ftr2              11
dtype: int64
```

```
print('군집 확인 후 적중', round(14/ 29 * 100),2, '%')
```

```
군집 확인 후 적중 48 2 %
```

심화

“ 내가 회사의 오너라면, 마케팅적인 관점에서 어떤 분석을 시행해야 하는가? ”

1. 최근에 구매하지 않은 고객을 재구매로 유도할 수 있는 방안을 무엇일까?
   (How do we induce buyers who have not recently purchased to repurchase?)

2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?
   (What product buyers purchased, and how can we predict buyers tendency?)

심화 1. 최근에 구매하지 않은 고객을 재구매로 유도할 수 있는 방안을 무엇일까?

구매 횟수가 제일 적은 고객을 찾는다. ID가 (12445, 12548, 12547, 12450, 12551)인 고객에 한해서 할인 쿠폰을 전송해 재구매를 유도할 수 있다.

```
repurchase = cust_df.sort_values(['Frequency'], ascending=True)
```

```
repurchase.head(5)
```

	CustomerID	Recency	Frequency	Monetary	Grade	cluster_label	Recency_log	Frequency_log	Monetary_log
4	12445	3171	4	133.40	Diamond	2	8.062118	1.609438	4.900820
22	12548	3315	5	95.20	Diamond	1	8.106515	1.791759	4.566429
21	12547	3346	8	207.80	Diamond	1	8.115820	2.197225	5.341377
6	12450	3305	8	197.88	Diamond	1	8.103494	2.197225	5.292702
24	12551	3506	10	168.00	Diamond	1	8.162516	2.397895	5.129899

심화 2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?

- 1. 고객들이 구매한 상품의 설명 Description을 분석해 소비 패턴을 파악한다.

```
retail_df.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	sale_amount
6421	536944	22383	LUNCH BAG SUKI DESIGN	70	2010-12-03 12:20:00	1.65	12557	Spain	115.5
6422	536944	22384	LUNCH BAG PINK POLKADOT	100	2010-12-03 12:20:00	1.45	12557	Spain	145.0
6423	536944	20727	LUNCH BAG BLACK SKULL.	60	2010-12-03 12:20:00	1.65	12557	Spain	99.0
6424	536944	20725	LUNCH BAG RED RETROSPOT	70	2010-12-03 12:20:00	1.65	12557	Spain	115.5
6425	536944	20728	LUNCH BAG CARS BLUE	100	2010-12-03 12:20:00	1.45	12557	Spain	145.0

심화 2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?

- 2. Postage, Frequency, Buy_in_one_time 컬럼을 추가해 한 고객이 몇 번 구매했는지 (Frequency), 구매 상품 중 배송비(postage) 결제 횟수를 파악하고, 한번 결제시 몇 개의 물건을 구매하였는지 파악한다. (Buy_in_one_time)

```
predict_tendency.loc[predict_tendency['Postage'] != 0, 'Buy_in_one_time'] = round(predict_tendency['Frequency'] / predict_tendency['Pos
predict_tendency.loc[predict_tendency['Postage'] == 0, 'Buy_in_one_time'] = 0
predict_tendency
```

```
C:₩Users₩USER₩anaconda3₩lib₩site-packages₩pandas₩core₩indexing.py:965: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a
-copy
  self.obj[item] = s
```

	CustomerID	Shopping_list	Postage	Frequency	Buy_in_one_time
4	12445	['BOXED CANDLES', 'BELL', 'BOXED CANDLES', 'POSTAGE']	1.0	4	4.00
22	12548	['PAPER DOILIES', 'PAPER DOILIES', 'PAPER DOILIES', 'COOKIE CUTTERS', 'POSTAGE']	1.0	5	5.00
21	12547	['CANDY ASSORTED', 'CARNIVAL ASSORTED', 'GIFT BOXES', 'SHAPE CUP', 'SHAPE CUP', 'POSTAGE', 'NAPKINS', 'POSTAGE']	2.0	8	4.00
6	12450	['DINER ASSORTED', 'GROCERY MAGNETS', 'DAY MAGNETS', 'JAR', 'CHOCOLATE CUPCAKES', 'PATTERN', 'PATTERN', 'TEA MUG']	0.0	8	0.00
24	12551	['BLACK WHITE', 'RED RETROSPOT', 'PINK POLKADOT', 'SCANDINAVIAN PAISLEY', 'WOODLAND ANIMALS', 'SHOPPER BAG', 'SHOPPER BAG', 'SHOPPER BAG', 'SHOULDER BAG', 'SHOULDER BAG']	0.0	10	0.00

심화 2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?

- 물건을 구매했지만 배송비(postage)를 결제한 이력이 없는 고객이 존재했습니다.

만약, Postage를 결제한 이력이 없다면 무료배송 쿠폰을 사용했을 것이라 추정하고, 라벨링을 더해줬습니다.

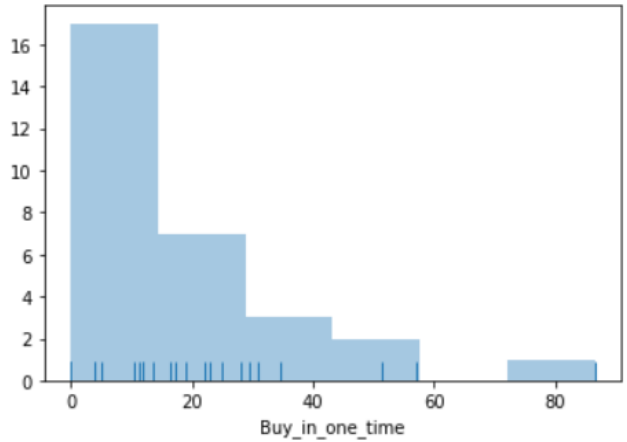
6	12450	['DINER ASSORTED', 'GROCERY MAGNETS', 'DAY MAGNETS', 'JAR', 'CHOCOLATE CUPCAKES', 'PATTERN', 'PATTERN', 'TEA MUG']	0.0	8	FREE DELIVERY COUPON	{'DINER ASSORTED': 1, 'GROCERY MAGNETS': 1, 'DAY MAGNETS': 1, 'JAR': 1, 'CHOCOLATE CUPCAKES': 1, 'PATTERN': 2, 'TEA MUG': 1}	[('MAGNETS', 2), ('DINER', 1), ('ASSORTED', 1), ('GROCERY', 1), ('DAY', 1), ('CHOCOLATE', 1), ('CUPCAKES', 1), ('TEA', 1), ('MUG', 1)]	X

심화 2. 시각화 진행

How many things buy in one purchase

```
sns.distplot(predict_tendency['Buy_in_one_time'], kde=False, rug=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x207d10d1888>
```

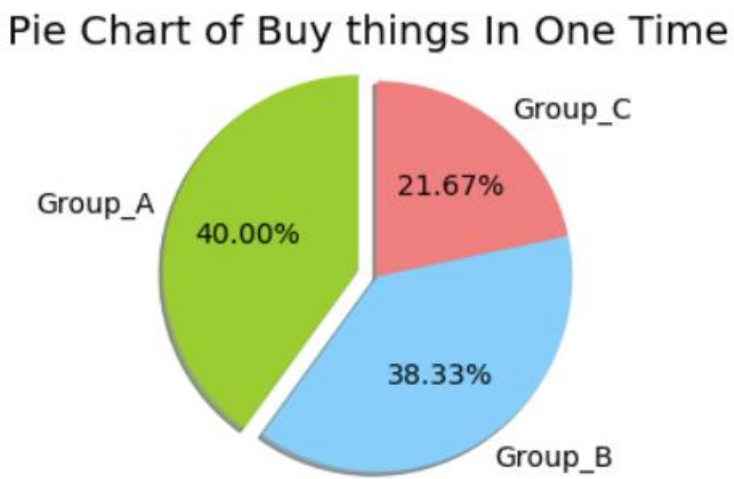



```
group_A = 0
group_B = 0
group_C = 0
predict_tendency[['Buy_in_one_time']].describe()
```

	Buy_in_one_time
count	30.000000
mean	17.103333
std	20.128694
min	0.000000
25%	0.000000
50%	12.000000
75%	24.500000
max	86.500000

심화 2. 시각화 진행 ; 한 번에 구매한 상품의 개수

- Group_A : 10개 이하로 상품을 구매한 고객의 수
- Group_B : 10개 초과 30개 이하 구매한 고객의 수
- Group_C : 31개 이상 구매한 고객의 수



-> 대부분의 고객이 1회에 10개 이상 구매를 하였다. 31개 이상 구매한 고객은 도소매업자라고 추정할 수 있었다.

심화 2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?

- 3. 자연어분석을 배우지 못했던 시점이지만, 최대한 분석할 수 있는 방법을 찾아 노력했습니다. 고객이 구매한 상품의 정보가 담긴 Description을 문자열 분석을 위한 함수를 만들었고, Wordcount를 진행했습니다.

```
def isOnlyChar(a) :
        result = ''
        for i in range(len(a)):
            if a[i] == ' ':
                if a[i-1].isalpha():
                    if a[i+1].isalpha():
                        result += ' '
            if a[i].isalpha():
                result += a[i]
        return result
```

```
for i in predict_tendency.index:
    tt = predict_tendency['Shopping_list'][i]
    li = {}
    li2 = {}

    tt2 = tt.split(',')
    for j in range(len(tt2)) :
        p = isOnlyChar(tt2[j])
        if p in li :
            li[p] += 1
        else :
            li[p] = 1

        if ' ' in p :
            p1 = p.split(' ')
            if p1[0] in li2:
                li2[p1[0]] += 1
            else :
                li2[p1[0]] = 1
            if p1[1] in li2:
                li2[p1[1]] += 1
            else :
                li2[p1[1]] = 1
    predict_tendency['Classify_Shopping_list'][i] = li
    dicArr = sorted(li2.items(), key=lambda x: x[1], reverse=True)
    predict_tendency['Frequency_Shopping_list'][i] = str(dicArr)
```

심화 2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?

- Classify_Shopping_list 와 Frequency_Shopping_list 컬럼을 추가해 고객이 구매한 Shopping_list를 카운팅합니다. 고객 분석을 위해서는 단어 하나하나를 분석하는 것이 필요하다고 생각했습니다. 이러한 이유로 파생변수 Frequency_Shopping_list을 생성해 단어 별로 쪼개 카운팅 작업을 진행했습니다. 그 후, 내림차순으로 정렬해 한 눈에 빈도를 확인 할 수 있도록 만들었습니다.

	CustomerID	Shopping_list	Postage	Frequency	Buy_in_one_time	Classify_Shopping_list	Frequency_Shopping_list
4	12445	['BOXED CANDLES', 'BELL', 'BOXED CANDLES', 'POSTAGE']	1.0	4	4.0	{'BOXED CANDLES': 2, 'BELL': 1, 'POSTAGE': 1}	[('BOXED', 2), ('CANDLES', 2)]
22	12548	['PAPER DOILIES', 'PAPER DOILIES', 'PAPER DOILIES', 'COOKIE CUTTERS', 'POSTAGE']	1.0	5	5.0	{'PAPER DOILIES': 3, 'COOKIE CUTTERS': 1, 'POSTAGE': 1}	[('PAPER', 3), ('DOILIES', 3), ('COOKIE', 1), ('CUTTERS', 1)]
21	12547	['CANDY ASSORTED', 'CARNIVAL ASSORTED', 'GIFT BOXES', 'SHAPE CUP', 'SHAPE CUP', 'POSTAGE', 'NAPKINS', 'POSTAGE']	2.0	8	4.0	{'CANDY ASSORTED': 1, 'CARNIVAL ASSORTED': 1, 'GIFT BOXES': 1, 'SHAPE CUP': 2, 'POSTAGE': 2, 'NAPKINS': 1}	[('ASSORTED', 2), ('SHAPE', 2), ('CUP', 2), ('CANDY', 1), ('CARNIVAL', 1), ('GIFT', 1), ('BOXES', 1)]
6	12450	['DINER ASSORTED', 'GROCERY MAGNETS', 'DAY MAGNETS', 'JAR', 'CHOCOLATE CUPCAKES', 'PATTERN', 'PATTERN', 'TEA MUG']	0.0	8	0.0	{'DINER ASSORTED': 1, 'GROCERY MAGNETS': 1, 'DAY MAGNETS': 1, 'JAR': 1, 'CHOCOLATE CUPCAKES': 1, 'PATTERN': 2, 'TEA MUG': 1}	[('MAGNETS', 2), ('DINER', 1), ('ASSORTED', 1), ('GROCERY', 1), ('DAY', 1), ('CHOCOLATE', 1), ('CUPCAKES', 1), ('TEA', 1), ('MUG', 1)]
24	12551	['BLACK WHITE', 'RED RETROSPOT', 'PINK POLKADOT', 'SCANDINAVIAN PAISLEY', 'WOODLAND ANIMALS', 'SHOPPER BAG', 'SHOPPER BAG', 'SHOPPER BAG', 'SHOULDER BAG', 'SHOULDER BAG']	0.0	10	0.0	{'BLACK WHITE': 1, 'RED RETROSPOT': 1, 'PINK POLKADOT': 1, 'SCANDINAVIAN PAISLEY': 1, 'WOODLAND ANIMALS': 1, 'SHOPPER BAG': 3, 'SHOULDER BAG': 2}	[('BAG', 5), ('SHOPPER', 3), ('SHOULDER', 2), ('BLACK', 1), ('WHITE', 1), ('RED', 1), ('RETROSPOT', 1), ('PINK', 1), ('POLKADOT', 1), ('SCANDINAVIAN', 1), ('PAISLEY', 1), ('WOODLAND', 1), ('ANIMALS', 1)]

심화 2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?

- 4. predict_customers_tendency 컬럼을 추가해 상품 이름에 크리스마스가 들어가는 물건을 구매하는 고객은 ‘Christmas Buyer’ 라는 라벨링을 더해주고, 고객이 선호하는 색상을 찾아 고객 특성에 라벨링을 더해준다.

```
colors = ['BLACK', 'RED', 'IVORY', 'WHITE', 'PINK', 'BLUE', 'GREEN', 'YELLOW', 'ORANGE', 'PURPLE', 'MINT', 'BROWN']
index = predict_tendency.index

for z in range(30) :
    dics = dict(di[z])
    arr = dics.keys()
    arrs = list(arr)
    strs = ''
    for i in arr:
        for j in colors:
            if j.find(i) != -1 :
                strs += j + ' '
    print('z = ', z , ' and color is ' , strs)
```

심화 2. 고객들은 어떤 상품들을 구매했고, 이를 통해 소비 성향을 파악 할 수 있을까?

```
    # if christmas is in top 5, result bought christmas things
    flag = 0
    five_tops = arrs[:5]
    if len(five_tops) > 1 :
        if arrs[0].find('CHRISTMAS') != -1 :
            flag = 1
    if len(five_tops) > 2 :
        if arrs[1] == 'CHRISTMAS' :
            flag = 1
    if len(five_tops) > 3 :
        if arrs[2] == 'CHRISTMAS' :
            flag = 1
    if len(five_tops) > 4 :
        if arrs[3] == 'CHRISTMAS' :
            flag = 1
    if len(five_tops) == 5 :
        if arrs[4] == 'CHRISTMAS' :
            flag = 1

    if flag == 1 :
        strs += ' CHRISTMAS BUYER '
    if len(strs) != 0 :
        ix = index[z]
        predict_tendency.loc[index==ix,  'predict_customers_tendency'] = strs
predict_tendency.head()
```

predict_customers_tend
PINK GREEN RED WHITI
IVORY MINT PINK MINT
BLACK CHRISTMAS BUY

총평

- 군집화가 주목적이었지만 나아가 비즈니스적인 관점에서 필요한 분석은 무엇인지 생각해보고, 고객을 분석하는 좋은 시도였습니다.

- 당시 Text Mining을 습득하지 못한 상황이라 혼자 최대한 할 수 있는 한에서 분석을 시도한 것이라 약간의 아쉬움이 남는 프로젝트였습니다. 하지만 앞으로 분석에 있어서 어떤 방향으로 분석을 해야하는지 알 수 있었던 프로젝트였습니다.

- END -