PORTFOLIO

이름	채 주 형
생년월일	1995. 02. 11
이메일	cown0211@naver.com
전화번호	010-2262-8170

PORTFOLIO
CONTENTS

01.	K-Means 클러스터링을 통한 고객 군집화 및 해석	고객의 신상 정보(생년, 학력, 자녀유무, 결혼유무 등)와 구매 정보(분야별 소비액수, 유입 경로 등)가 담긴 데이터로부터 고객을 군집화하여 이를 각 군집에 대한 해석을 진행한 프로젝트
02.	정보시스템 설계	웹크롤링, 반응형차트, Dash 웹페이지를 활용하여 데이터베이스의 정보를 시각화하는 프로젝트
03.	소득 분위에 대한 소득, 지출 분석	소득 분위에 따른 지출 분야의 차이를 회귀해석을 적용하여 분석 및 이로부터 발생하는 불균형에 대한 해석
04.	진단 정보를 통한 치매 예측 모형	치매로 의심되는 환자들의 MRI 분석 결과를 이용하여 뇌부피 측정 및 기억력 검사 등을 통해 치매 여부를 판별하는 모형 설계
05.	분리불안 의심 아동 행동분석 Ai를 통한 진찰 권고 시스템	아동의 행동이 담긴 영상으로부터 아동의 행동이 분리불안이 의심되는지 여부를 반환하는 모형 개발

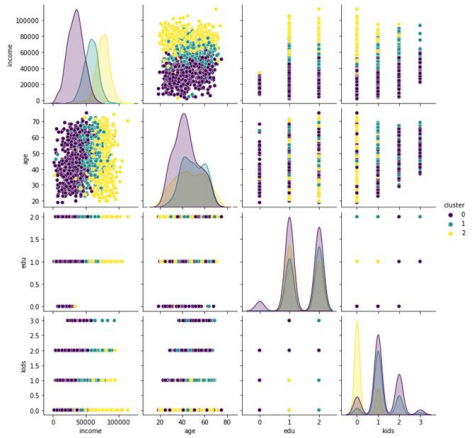
Project

K-Means 클러스터링을 통한 고객 군집화 및 해석

About project

고객의 신상 정보(생년, 학력, 자녀유무, 결혼유무 등)와 구매 정보(분야별 소비액수, 유입 경로 등)가 담긴 데이터로부터 고객을 군집화하여 이를 각 군집에 대한 해석을 진행한 프로젝트

K-Means 클러스터링을 통한 고객 군집화 및 해석

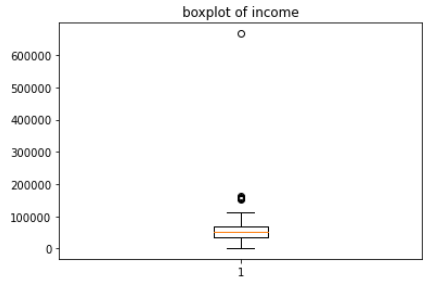


【그림 13】 개인정보에 따른 그부

【그림 13】으로부터 그룹0, 1, 2로 갈수록 소득이
또한, 나이대 분포에서 상대적으로 그룹0은 젊고,
간으로 볼 수도 있으나 분포상의 큰 차이라고 보

In [12]:
```
# birthyear 이상치 제거
cust_birthyear = custdf[(custdf['birthyear'] < 1930)]
custdf = custdf[~(custdf['birthyear'] < 1930)]
```

In [13]:
```
# income 이상치 탈색
plt.boxplot(custdf['income'])
plt.title('boxplot of income')
plt.show()
custdf.loc[custdf['income'] > 300000,:]
```



Introduce project

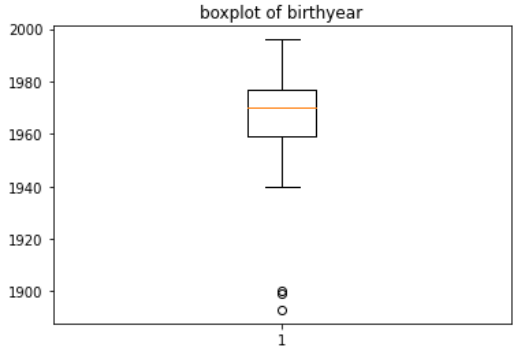
작업 기간	2021. 10 ~ 2021. 12 (3개월)
인력 구성(기여도)	개인 프로젝트
프로젝트 목적	군집화 및 시각화 역량 강화
프로젝트 내용	주어진 데이터로부터 고객을 군집화 하여 이를 각 군집에 대한 해석을 진행한 프로젝트
주요 업무 및 상세 역할	1) 데이터 전처리(날짜 -> 수치, 새로운 파생 변수로 통합, 결측치 및 이상치 처리) 2) 군집화 및 시각화
사용언어 및 개발 환경	Python, Jupyter Notebook
참고 자료	Kaggle-DACON/marketing_campagin.ipynb at main · cown0211/Kaggle-DACON · GitHub

Main work 데이터 전처리

- **이상치 및 결측치 제거**

  전체 데이터 개수에 비해 이상치 및 결측치의 비중이 매우 낮은 것을 고려하여 제거하고 분석

In [11]:
```
# birthyear 이상치 탐색
plt.boxplot(custdf['birthyear'])
plt.title('boxplot of birthyear')
plt.show()
custdf[(custdf['birthyear'] < 1930)]
```



- **모든 레코드에 대해 똑같은 값을 갖는 의미 없는 변수 제거**

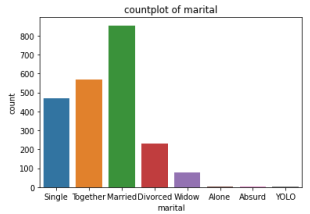
- **'생년' 데이터를 직관적이도록 나이로 변환**
- **아동,청소년 자녀수 변수는 합쳐 자녀수로 변환**
- **각 캠페인에 응한 횟수는 모두 더해 캠페인에 응한 총 횟수로 변환**

In [22]:
```
# 변수 생성
# age
custdf['age'] = today.year - custdf['birthyear']

# kids
custdf['kids'] = custdf['kid'] + custdf['teen']

# totalcmp
custdf['totalcmp'] = custdf['cmp1'] + custdf['cmp2'] + custdf['cmp3'] + custdf['cmp4'] + custdf['cmp5']
```

- **'싱글','동거','이혼','사별' 등의 값을 갖는 결혼여부 변수는 배우자 유무로 변환**

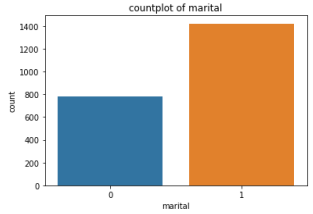
In [19]:
```
# countplot of marital
sns.countplot(custdf['marital'])
plt.title('countplot of marital')
plt.show()
```

```
C:\Users\vngud\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [20]:
```
# marital 변수, 2개 범주로 변환
custdf['marital'].replace(['Single', 'Divorced', 'Widow', 'Alone', 'Absurd', 'YOLO'], 0, inplace=True)
custdf['marital'].replace(['Together', 'Married'], 1, inplace=True)
sns.countplot(custdf['marital'])
plt.title('countplot of marital')
plt.show()
```

```
C:\Users\vngud\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



K-Means 클러스터링을 통한 고객 군집화 및 해석

Main work 군집화, 시각화, 해석

- **군집의 개수를 정하기 위해 ELBOW METHOD 활용**

  군집의 개수는 3

- **고객 데이터를 4가지 카테고리로 분류하여 각 군집에 대한 해석을 진행**

  개인정보,구매상품,유입경로,캠페인민감도에 따른 분류

- **각 그룹에 대한 해석(0그룹, 1그룹, 2그룹)**

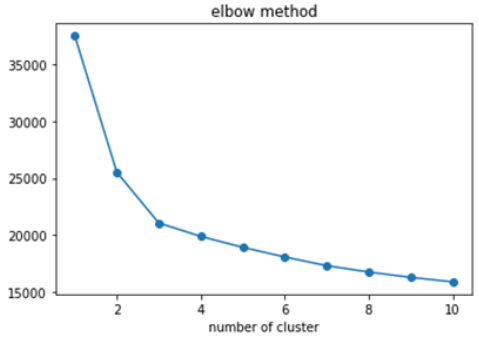
  0그룹; 상대적 저소득, 저연령, 저학력, 자녀1~2명, 웹방문 잦으나 할인이나 캠페인에는 잘 응하지 않음

  1그룹; 중간 소득층, 고령, 고학력, 자녀1~2명, 오프라인 선호, 할인 구매 잦음

  2그룹; 고소득, 고른 나이 분포, 중간 학력, 자녀0~1명, 할인 및 웹사이트 방문 빈도 낮음

In [ ]:
```
### elbow method ###

# 데이터 스케일링
scaler = StandardScaler()
custdf_scaling = scaler.fit_transform(custdf.values)

# inertia 계산
tmp = []
for i in range(1, 11):
  kmeans = KMeans(n_clusters = i, max_iter = 1000)
  kmeans.fit(custdf_scaling)
  tmp.append(kmeans.inertia_)
  print('n_cluster', i, ':', kmeans.inertia_)

plt.plot(range(1, 11), tmp, marker = 'o')
plt.title('elbow method')
plt.xlabel('number of cluster')
plt.show()
```

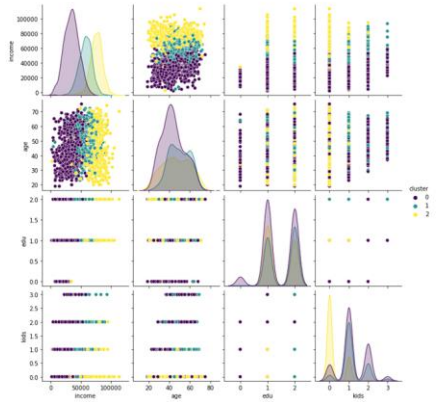
In [ ]:
```
# clustering, k = 3
kmean = KMeans(n_clusters = 3, max_iter = 1000)
kmean.fit(custdf_scaling)
kmean.labels_
custdf['cluster'] = kmean.labels_
```



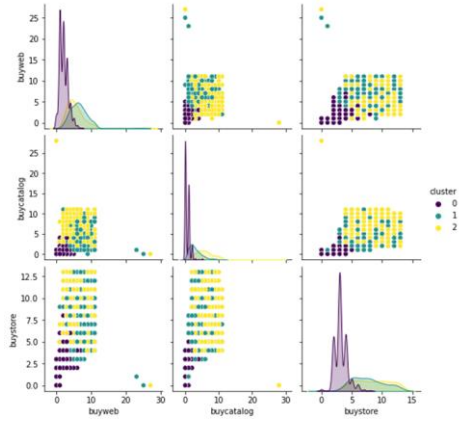
【그림 12】 elbow method

군집	고객수
0	1012
1	601
2	592

〈표 4〉 군집별 고객수



【그림 13】 개인정보에 따른 구분



【그림 15】 구매경로에 따른 구분

Project 02.

정보시스템 설계

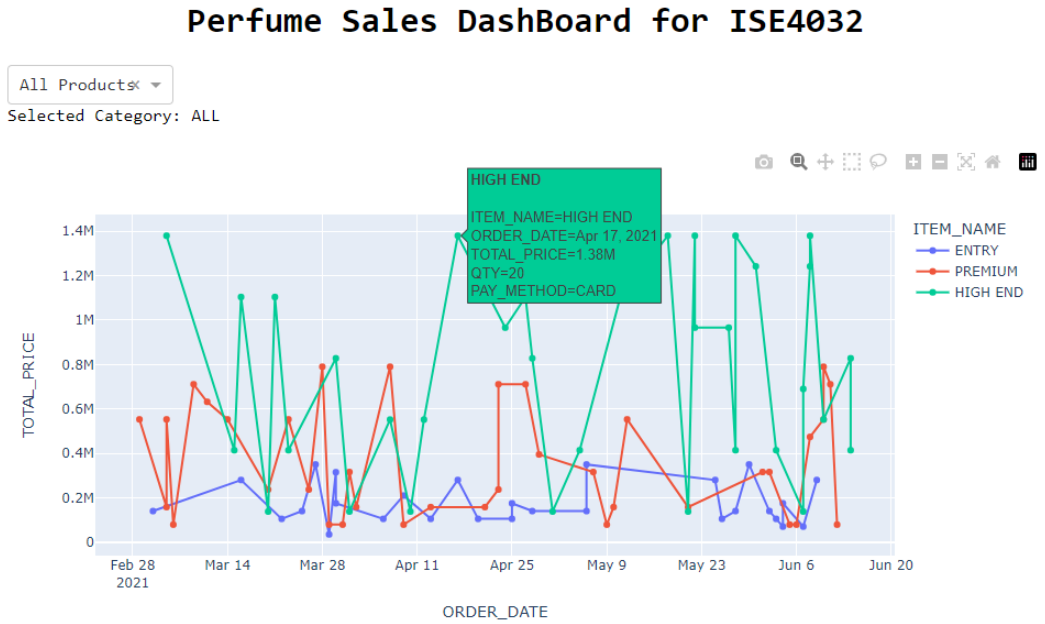
About project

웹크롤링, 반응형차트, Dash 웹페이지를 활용하여 데이터베이스의 정보를 시각화하는 프로젝트

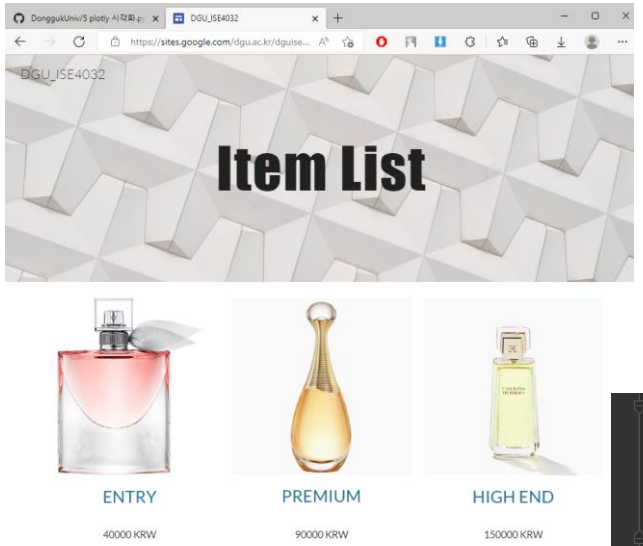
Project 02. 정보시스템 설계

Introduce project

작업 기간	2021.03~2021.06
인력 구성(기여도)	학과 개인 프로젝트
프로젝트 목적	파이썬을 활용하여 주어진 데이터를 시각화하고 변동사항을 반영하여 웹페이지를 통해 시각화
프로젝트 내용	- 주어진 데이터를 반응형 차트로 시각화 - 데이터의 변동사항을 웹크롤링을 통해 반영 - 웹페이지를 활용한 시각화
주요 업무 및 상세 역할	1) 반응형 차트 제작 2) 웹 크롤링을 통해 변동사항 반영 3) 웹페이지를 통해 시각화
사용언어 및 개발 환경	Python, PyCharm
참고 자료	DonggukUniv/21_1_InformationSystem_ISE at main · cown0211/DonggukUniv · GitHub



Main work 웹크롤링



- **Beautifulsoup 라이브러리를 활용하여 웹페이지의 텍스트를 모두 읽어온 후 프로젝트의 목적에 맞게 전처리 하는 과정을 거침**

- **아이템의 종류에 따른 가격을 가져오기 위한 목적으로 전처리 시행**

```python
import pandas as pd
import math
from bs4 import BeautifulSoup # Ctrl + Space beautifulsoup 선택
from urllib.request import urlopen
# 패키지 beautifulsoup4, requests 설치
from datetime import date # 기본 패키지

def load_data():
    html = urlopen('https://sites.google.com/dgu.ac.kr/dguise4032/itemlist')
    # 웹페이지 전체 컨텐츠를 긁어옴
    bsObject = BeautifulSoup(html, 'html.parser')
    # HTML 파일을 불러와서 파싱(=변환)
    # Variables 중 text 가면 웹 페이지의 text 다 긁어옴
    print(bsObject.text)
    print(bsObject.getText())
    # 후자가 함수로 원하는 값 반환받기 용이

    # 긁어오긴 하는데 여기서 가격을 어떻게 추출하나?
    text = bsObject.getText()
    print(text.index('KRW')) # 텍스트 내의 특정 키워드 위치 반환

    # 특정 키워드를 다른 키워드로 대체하고 이 전후로 슬라이싱
    text = text.replace('KRW', '#') # KRW -> #으로 변환
    text = text.replace('ENTRY', '#')
    text = text.replace('PREMIUM', '#')
    text = text.replace('HIGH END', '#')
```

Main work 반응형 차트

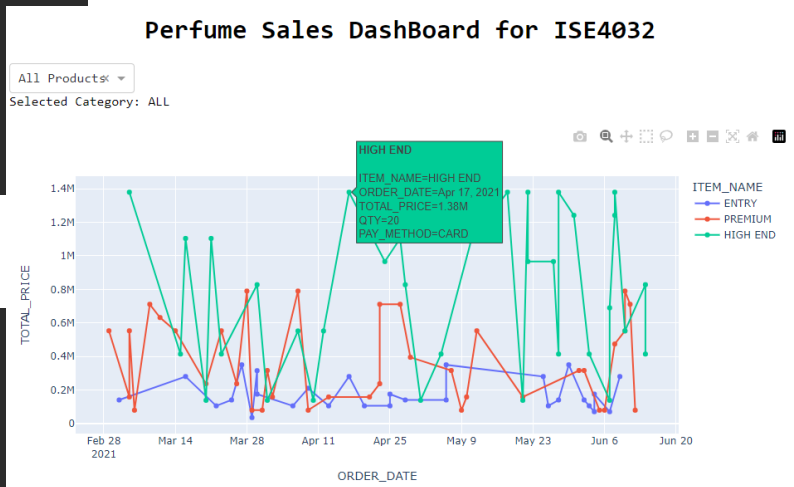
- **Plotly 라이브러리 활용**

- **차트의 점에 마우스를 갖다 대면 점이 갖고 있는 데이터를 표시해주는 반응형 차트 제작**


```
ucture.xlsx', sheet_name='ORDER')

Structure.xlsx', sheet_name='CUSTOMER')

ucture.xlsx', sheet_name='STOCK')

    # 새로운 Dataframe 생성 - ORDER와 STOCK을 조합하여 새 테이블로!
    df_new = df_stock.set_index('ITEM_ID').join(df_order.set_index('ITEM_ID'))  # STOCK 관련 정보가 앞으로 ORDER가 뒤로
    # df_new2 = df_order.set_index('ITEM_ID').join(df_stock.set_index('ITEM_ID')) # ORDER 앞으로 STOCK이 뒤로
    df_new = df_new.set_index('CUST_ID').join(df_customer.set_index('CUST_ID'))
    df_new['TOTAL_PRICE'] = df_new['PRICE'] * df_new['ITEM_QTY']  # Total Price라는 새로운 Column을 계산을 통해 생성
    df_new['FULL_NAME'] = df_new['CUST_LNAME'] + " " + df_new['CUST_FNAME']

    # 반응형 차트 생성
    fig = px.line(df_new, title="ISE4032 Sales Record", x='ORDER_DATE', y='TOTAL_PRICE', color='ITEM_NAME',
                  hover_name='FULL_NAME', hover_data=['QTY'])
    fig.update_traces(mode="markers")  # 마커 생성
    fig.show()
    print('Done')


if __name__ == '__main__':
    load_data()
```

Main work Dash 웹페이지 제작

```
app.layout = html.Div(
    [
        html.H1('Perfume Sales DashBoard for ISE4032',style={'text-align':'center','font-family':'consolas'}),
        # .H1()은 글자 크기! 제일 큰것, H2,3,4~~~도 있음                중앙정렬              폰트는 콘솔라스
        # ㄴ타이틀
        dcc.Dropdown(
            id='select_option',
            options=[
                {'label':'Entry','value':'ENTRY'},
                {'label':'Premium','value':'PREMIUM'},
                {'label':'High End','value':'HIGH END'},
                {'label':'All Products','value':'ALL'}
            ],
            multi=False,   # 다중선택 불가!
            value='ALL',    # 기본값 ALL로 설정 , 여기까지만 하면 넙대대...
            style={'width':'40%','font-family':'consolas'}
        ),
        html.Div(id='output_container',children=[],style={'font-family':'consolas'}),
        # ㄴ자르에 대한 옵션
        html.Br(),  # 줄바꿈
        dcc.Graph(id='sales_info',figure={}) # 차트그려줌
    ]
```



- Dash 라이브러리를 활용하여 앞서 만든 반응형 차트를 웹페이지에 띄우는 형식으로 시각화 함

- 주어진 데이터에 따라 모든 ITEM을 볼 수 있고, 카테고리를 구분하여 볼 수 있음

```
@app.callback( # 콜백함수를 이용하여 원하는 옵션에 맞게 생성된 차트를 다시 레이아웃으로 반환
    [
        Output(component_id='output_container',component_property='children'),
        Output(component_id='sales_info',component_property='figure')

    ],
    [Input(component_id='select_option',component_property='value')]
)



def update_graph(option_selected):   # 선택된 옵션대로 차트 업데이트 위함!
    container = 'Selected Category: {}'.format(option_selected) # 선택된

    if option_selected == 'ALL':
        df_new = df_stock.set_index('ITEM_ID').join(df_order.set_index('ITEM_ID'))
        df_new['TOTAL_PRICE'] = df_new['PRICE'] * df_new['ITEM_QTY']
    else:
        df_new = df_stock.set_index('ITEM_ID').join(df_order.set_index('ITEM_ID'))
        df_new['TOTAL_PRICE'] = df_new['PRICE'] * df_new['ITEM_QTY']
        df_new = df_new.loc[df_new['ITEM_NAME'] == option_selected]  # 원하는 ITEM만 가져옴

    fig = px.line(df_new, x='ORDER_DATE', y='TOTAL_PRICE', color='ITEM_NAME',
                  hover_name='ITEM_NAME', hover_data=['QTY','PAY_METHOD'])
```

Project 03.

소득 분위에 대한 소득, 지출 분석

About project

소득 분위에 따른 지출 분야의 차이에 대한 분석 및 이로부터 발생하는 불균형에 대한 해석

Project 03. 소득 분위에 대한 소득, 지출 분석

Introduce project

작업 기간	2020.03~2020.06
인력 구성(기여도)	학과 팀 프로젝트(2인)
프로젝트 목적	소득 분위에 따른 지출 분야의 차이에 대한 분석 및 이로부터 발생하는 불균형에 대한 해석
프로젝트 내용	- 소득 분위 별 지출 분야의 차이에 대한 분석 - 소득 분위 별 저축불균형에 대한 비교
주요 업무 및 상세 역할	1) 회귀식 작성, 유의수준 및 다중공선성 분석 2) 소득분위 별 지출분야 비교
사용언어 및 개발 환경	R
참고 자료	DonggukUniv/회귀분석 프로젝트 코드 최종.r at main · cown0211/DonggukUniv · GitHub

*그림1-2. 소득분위별 연도에 따른 소득과 지출의 변화

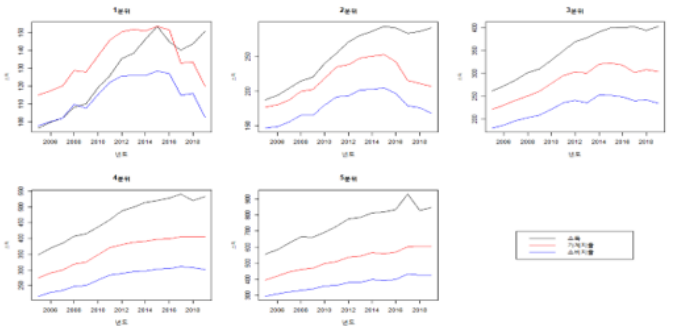
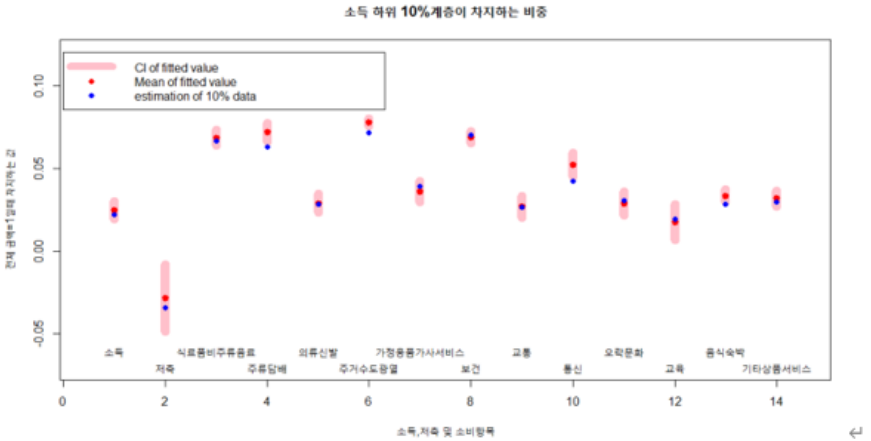


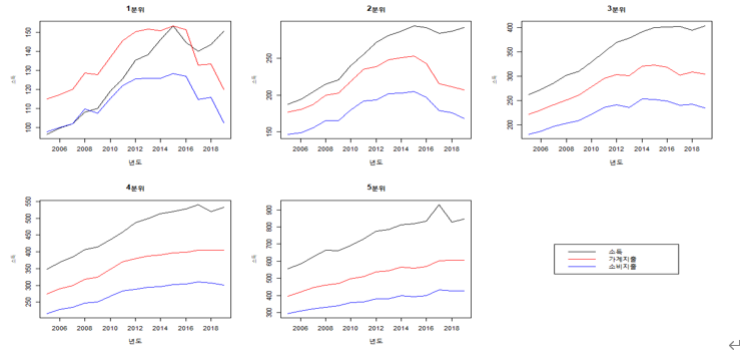
*그림6-1. 하위 10%계층이 차지하는 비중 추정



Main work 소득분위, 연도별 지출분야 분석

- **X축을 연도로, Y축을 액수로 하는 차트를 소득분위 별로 구분하여 시각화함으로 소득분위 별로 연도에 따라 지출과 소득 총액에서 보이는 추세가 다른 것을 확인함**

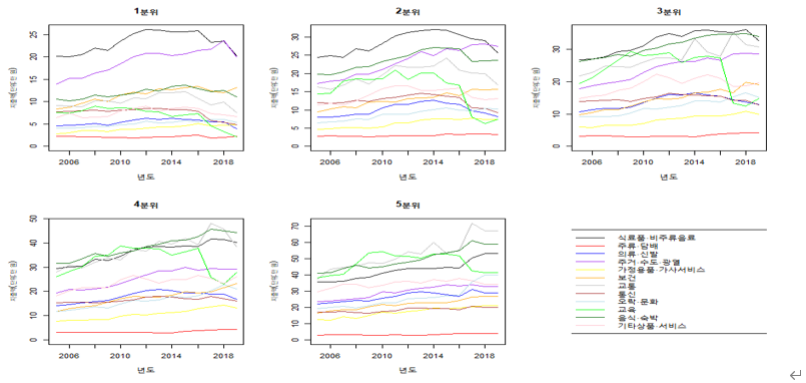
```
#그림1-2. 년도에 따른 소득,가계지출,소비지출의 변화
#소비지출=가계지출-비소비지출
#비소비지출= 조세,공적연금,사회보험,비영리단체로이전, 가구간 이전 등
par(mfrow=c(2,3))
plot(소득~년도,type="l",data=c_1,main='1분위',ylim=c(min(소득,가계지출,소비지출),max(소득,가계지출,소비지출)))
lines(가계지출~년도,type="l",col="Red",data=c_1)
lines(소비지출~년도,type="l",col="blue",data=c_1)
plot(소득~년도,type="l",data=c_2,main='2분위',ylim=c(min(소득,가계지출,소비지출),max(소득,가계지출,소비지출)))
lines(가계지출~년도,type="l",col="Red",data=c_2)
lines(소비지출~년도,type="l",col="blue",data=c_2)
plot(소득~년도,type="l",data=c_3,main='3분위',ylim=c(min(소득,가계지출,소비지출),max(소득,가계지출,소비지출)))
lines(가계지출~년도,type="l",col="Red",data=c_3)
lines(소비지출~년도,type="l",col="blue",data=c_3)
plot(소득~년도,type="l",data=c_4,main='4분위',ylim=c(min(소득,가계지출,소비지출),max(소득,가계지출,소비지출)))
lines(가계지출~년도,type="l",col="Red",data=c_4)
lines(소비지출~년도,type="l",col="blue",data=c_4)
plot(소득~년도,type="l",data=c_5,main='5분위',ylim=c(min(소득,가계지출,소비지출),max(소득,가계지출,소비지출)))
lines(가계지출~년도,type="l",col="Red",data=c_5)
lines(소비지출~년도,type="l",col="blue",data=c_5)
plot.new();legend("center",legend=c("소득","가계지출","소비지출"),col=c('black','red','blue'),lty=1)
###소득은 분위에 상관없이 증가하는 추세이나, 지출은 저분위일수록 증가->감소 추세이고 고분위일수록 증가추세이다.
```



- **각 소비분야에 따르는 추세를 정확히 확인하고자 소비분야에 따라 다른 색을 입혀 시각화 함**

  여러 차이가 존재했지만 가장 논에 띄는 것은 저분위일수록 주거/수도/광열 분야로의 지출이, 고분위일수록 교육, 음식/숙박 분야로의 지출이 커지는 것을 확인

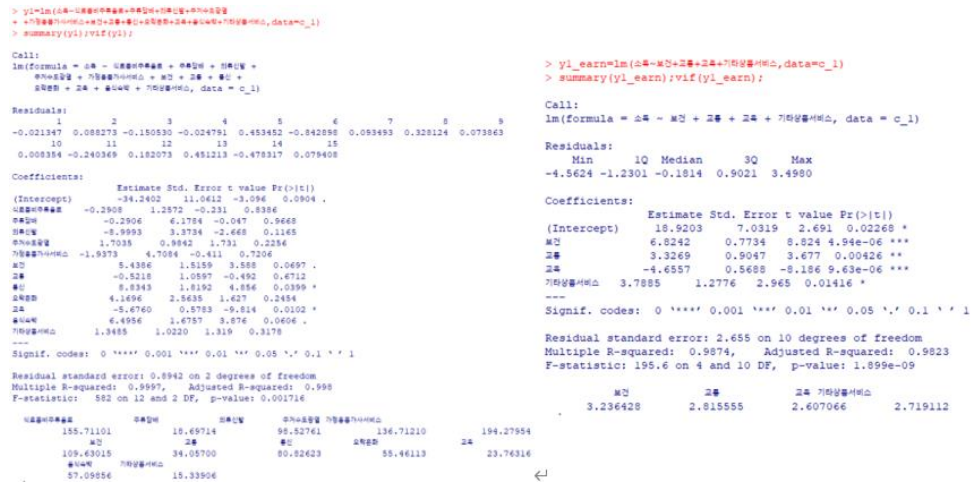
```
#그림2-2. 년도에 따른 소비분야별 액수변화
par(mfrow=c(2,3))
plot(c_1[,6]~c_1[,1],type="l",ylim=c(0,max(c_1[,6:17])),xlab='년도',ylab='지출액(단위:만 원)',
lines(c_1[,7]~c_1[,1],type="l",col='red')
lines(c_1[,8]~c_1[,1],type="l",col='blue')
lines(c_1[,9]~c_1[,1],type="l",col='purple')
lines(c_1[,10]~c_1[,1],type="l",col='yellow')
lines(c_1[,11]~c_1[,1],type="l",col='orange')
lines(c_1[,12]~c_1[,1],type="l",col='grey')
lines(c_1[,13]~c_1[,1],type="l",col='brown')
lines(c_1[,14]~c_1[,1],type="l",col='light blue')
lines(c_1[,15]~c_1[,1],type="l",col='green')
lines(c_1[,16]~c_1[,1],type="l",col='dark green')
lines(c_1[,17]~c_1[,1],type="l",col='pink')
plot(c_2[,6]~c_2[,1],type="l",ylim=c(0,max(c_2[,6:17])),xlab='년도',ylab='지출액(단위:만 원)',
```



Main work 소득분위, 연도별 지출분야 분석

- **소득(총액)과 지출(총액)을 타겟으로 하고 각 지출분야를 변수로 하는 회귀식을 작성함**

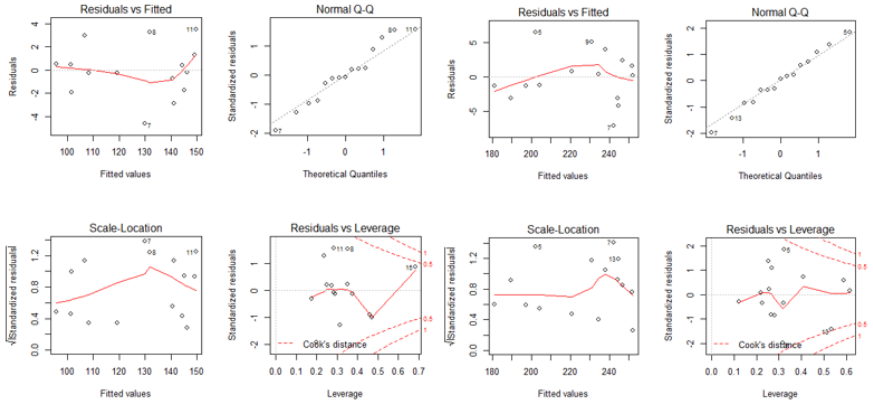
  각 소득분위 별로 회귀식을 각각 작성하여 총 10개의 회귀식을 작성함. 소득과 지출에 영향을 주는 요소를 파악하기 위해 회귀식의 변수의 유의성과 다중공선성을 파악하여 회귀식 작성



```
> yl_earn=lm(소득~보건+교통+교육+기타상품서비스,data=c_1)
> summary(yl_earn);vif(yl_earn);

Call:
lm(formula = 소득 ~ 보건 + 교통 + 교육 + 기타상품서비스, data = c_1)

Residuals:
    Min      1Q  Median      3Q     Max
-4.5624 -1.2301 -0.1814  0.9021  3.4980

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   18.9203     7.0319   2.691  0.02268 *
보건           6.8242     0.7734   8.824 4.94e-06 ***
교통           3.3269     0.9047   3.677  0.00426 **
교육          -4.6557     0.5688  -8.186 9.63e-06 ***
기타상품서비스  3.7885     1.2776   2.965  0.01416 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.655 on 10 degrees of freedom
Multiple R-squared:  0.9874,    Adjusted R-squared:  0.9823
F-statistic: 195.6 on 4 and 10 DF,  p-value: 1.899e-09

    보건     교통     교육 기타상품서비스
3.236428 2.815555 2.607066     2.719112
```

- **각 회귀모형에 대한 잔차분석 시행**

  엄격하게 잔차의 정규성을 따른다고 보기는 어려웠으나 전반적으로 정규성을 따른다고 가정하기에는 충분하였음



Project 04.

진단 정보를 통한 치매 예측 모형

About project

치매로 의심되는 환자들의 MRI 분석 결과를 이용하여 뇌부피 측정 및 기억력 검사 등을 통해 치매 여부를 판별하는 모형 설계

Project 04. 진단 정보를 통한 치매 예측 모형

Introduce project

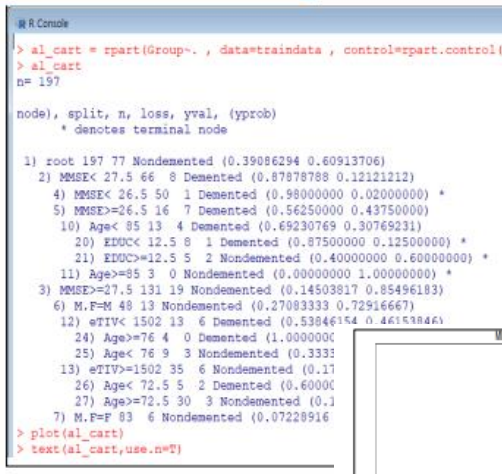
작업 기간	2019.09~2019.12
인력 구성(기여도)	학과 팀 프로젝트(5인)
프로젝트 목적	치매 환자 여부를 판별하는 적절한 모형을 개발
프로젝트 내용	- Logistic Regression, Decision Tree, Random Forest, KNN, Naïve Bayes, CART 모형을 활용하여 가장 적절한 예측모형을 찾아내고 치매 판별에 가장 큰 영향을 주는 변수를 찾아냄
주요 업무 및 상세 역할	1) 모델링 이후 변수의 유의성 분석 2) 모델 간 정확도 비교
사용언어 및 개발 환경	R
참고 자료	DonggukUniv/19_2_DataMining_ISE/proj at main · cown0211/DonggukUniv · GitHub

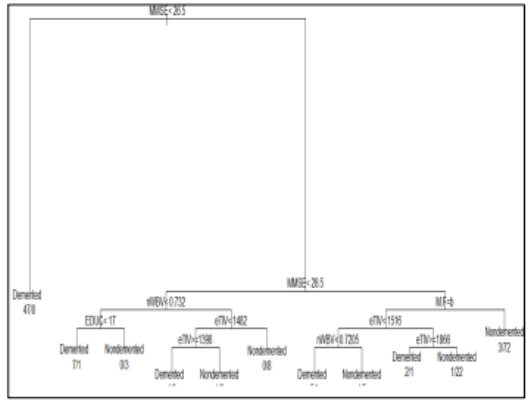
Project 04. 진단 정보를 통한 치매 예측 모형

Main work

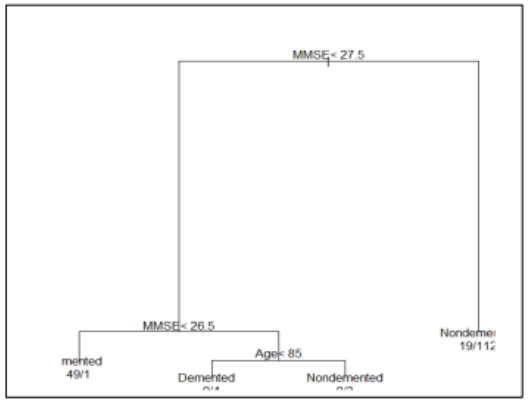
- 주어진 데이터에 Logistic Regression, Decision Tree, Random Forest, KNN, Naïve Bayes, CART 모형을 적용시켜 각 모형에서의 변수의 유의성을 비교하여 치매 진단에 가장 중요한 변수를 파악함

- 각 모델에서의 정확도는 70% ~ 87%로 모두 달랐으나 분류에 있어 가장 중요한 변수는 모두 MMSE라는 변수를 나타냈음.





Tree1. Pruning 실시 전



Tree1. Pruning 실시 후

Project 05.

분리불안 의심 아동 행동분석 AI를 통한 진찰 권고 시스템

About project

아동의 행동이 담긴 영상으로부터 아동의 행동이 분리불안이 의심되는지 여부를 반환하는 모형 개발

Project 05. 분리불안 의심 아동 행동분석 AI를 통한 진찰 권고 시스템

Introduce project

작업 기간	2021.01~2021.05
인력 구성(기여도)	대외 팀 프로젝트(5인)
프로젝트 목적	영상 데이터를 정형화 하고 정형화된 데이터를 통해 분리불안이 의심되는지 여부를 판별하는 모형 개발
프로젝트 내용	- 아동의 행동이 담긴 영상으로부터 아동의 행동이 분리불안이 의심되는지 여부를 반환하는 모형 개발
주요 업무 및 상세 역할	1) 영상 전처리(1초 단위로 자름) 2) 이미지에 CNN, openCV 모형 적용 3) 반환된 정형 데이터에 XGBoost 모형 적용
사용언어 및 개발 환경	Python, Jupyter Nootebook
참고 자료	

Main work

- **아동의 행동이 담긴 영상을 1초 단위의 이미지로 자른 뒤, 각 이미지에 기존에 정의한 4가지 행동값, 4가지 감정값을 할당함**

- **행동값이 부여된 이미지에는 openCV를, 감정값이 부여된 이미지에는 CNN 알고리즘을 적용하여 훈련시킨 뒤 이미지가 어떤 행동 or 감정값을 가지게 되는지 1차 예측 모형을 설계**

  이때 알고리즘이 내놓는 아웃풋은 행동, 감정값을 attribute로 갖는 행렬 형태

```python
from PIL import Image
import os, glob, numpy as np
from keras.models import load_model

caltech_dir = "picturedata3/분리불안 아닌 데이터/data (1)"
image_w = 28
image_h = 28

pixels = image_h * image_w * 3

X = []
filenames = []
files = glob.glob(caltech_dir+"/*.*")
for i, f in enumerate(files):
    img = Image.open(f)
    img = img.convert("RGB")
    img = img.resize((image_w, image_h))
    data = np.asarray(img)
    filenames.append(f)
    X.append(data)

X = np.array(X)
model = load_model('Gersang.h5')

prediction = model.predict(X)
np.set_printoptions(formatter={'float': lambda x: "{0:0.3f}".form
cnt = 0
e, f, g, h =0, 0, 0, 0

for i in prediction:
    pre_ans = i.argmax()  # 예측 레이블
    print(i)
    print(pre_ans)
    pre_ans_str = ''
    if pre_ans == 0: pre_ans_str = "cry"
    elif pre_ans == 1: pre_ans_str = "sad"
    elif pre_ans == 2: pre_ans_str = "mad"
    elif pre_ans == 3: pre_ans_str = "innocence"
```

In [ ]:

```python
caltech_dir = "c://data/분리불안/9"
image_w = 64
image_h = 64

pixels = image_h * image_w * 3

X = []
filenames = []
files = glob.glob(caltech_dir+"/*.*")
for i, f in enumerate(files):
    img = Image.open(f)
    img = img.convert("RGB")
    img = img.resize((image_w, image_h))
    data = np.asarray(img)
    filenames.append(f)
    X.append(data)

X = np.array(X)
model = load_model('./multiimg_classification.model')
```

```python
del.predict(X)
ions(formatter={'float': lambda x: "{0:0.3f}".format(x)})

0, 0, 0
```

	data	cry	sad	mad	innocence	begging	deny	stable	violence	anxiety
0	data (1)	0.0	17.0	0.0	82.0	0.0	0.0	71.0	29.0	0.0
1	data (2)	4.0	31.0	6.0	59.0	0.0	3.0	64.0	33.0	0.0
2	data (3)	1.0	18.0	0.0	80.0	0.0	6.0	49.0	44.0	0.0
3	data (4)	3.0	12.0	8.0	75.0	1.0	16.0	41.0	42.0	0.0
4	data (5)	0.0	12.0	0.0	87.0	0.0	6.0	13.0	81.0	0.0
5	data (6)	3.0	5.0	25.0	64.0	5.0	21.0	27.0	47.0	0.0
6	data (7)	3.0	4.0	12.0	80.0	5.0	41.0	27.0	26.0	0.0
7	data (8)	4.0	11.0	16.0	68.0	3.0	22.0	32.0	43.0	0.0
8	data (9)	10.0	5.0	0.0	84.0	0.0	26.0	53.0	21.0	0.0
9	data (10)	15.0	12.0	40.0	31.0	1.0	30.0	43.0	27.0	0.0

Main work

- **이미지 분류기로부터 얻어낸 행렬 데이터에서 타겟변수는 분리불안 여부, 종속변수는 행동,감정 예측값으로 하는 최종 예측 모형 설계**

  분리불안 의심 = 1; 분리불안 의심x = 0

- **Logistic Regression, Decision Tree, Random Forest, XGBoost 모델에 각각 적용시켜 가장 정확도가 높은 XGBoost 모형을 최종적으로 선택**

  train set에서의 정확도는 Logistic Regression이 가장 높았으나, test set에서의 정확도는 XGBoost가 더 높았음

```
forest = RandomForestClassifier(n_estimators=100)
forest.fit(X_train, y_train)


y_pred = forest.predict(X_test)


print('정확도 :', metrics.accuracy_score(y_test, y_pred))
```

```
정확도 : 0.7073170731707317
```

```
print(classification_report(y_pred, y_tes
```

	precision	recall	f1-score
0.0	0.81	0.76	0.79
1.0	0.50	0.58	0.54
accuracy			0.71
macro avg	0.66	0.67	0.66
weighted avg	0.72	0.71	0.71

```
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
print(model.score(X_train, y_train))
```

```
0.7333333333333333
```

```
print(model.score(X_test, y_test))
```

```
0.7560975609756098
```

```
pred = model.predict(X_test)
```

```
confusion_matrix(y_test, pred)
```

```
2],
6]], dtype=int64)
```

```
b = confusion_matrix(y_test,xgb_pred)
```

```
b
```

```
array([[24,  3],
       [ 8,  6]], dtype=int64)
```

```
(b[0,0]+b[1,1])/(b[0,0]+b[1,0]+b[0,1]+b[1,1])
```

```
0.7317073170731707
```

End of Document