**국가금연지원 서비스**

**만족도 설문분석**

|  |
| --- |
|  |

**응용통계학과**

**주건재**

|  |
| --- |
|  |

<  목    차  >

1 서론 1

1.1분석개요 1

1.2 분석목표 2

1.3 데이터소개 2

1.4 분석 모델 3

2 본론 4

2.1 분석 플로우 4

2.2 EDA 및 전처리 5

2.3 RE-sampling 6

2.4 multicollinearity 제거 8

2.5 Correlation analysis 8

2.6 evaluation 방법 9

2.7 분석결과 및 토의 10

3 결론 13

3.1 연구의 결론 및 한계 13

3.2 추후연구 13

4 REFERENCES 14

5 Appendix 15

A 실제설문지 15

B source code 15

1. **서론**
   1. **분석개요**

한국건강증진개발원에서 매년 진행하는 “보건소 금연 클리닉 이용자 대상 만족도조사”의 설문조사 데이터를 기반으로 금연 클리닉 서비스 질의 개선방향을 리서치 하는 것이 분석의 목적이다.

-설문소개-  
조사목적: 국가금연지원서비스의 품질 및 근거 기반의 정책 마련 도모  
추진개요: (기간) 2021.8.9 ~ 조사완료 시까지  
조사방법: 설문지를 통한 전화조사  
설문대상: 서비스 이용 후 6개월 경과자 5000명

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 설문 내용 | 대분류 | 소분류 | 개선방법 |
| 1 | 금연상담사는 상담 약속시간을 잘 지켰습니까? | 상담사 | 서비스 | 상담사 교육 |
| 2 | 금연하는 동안 상담사로부터 도움을 충분히 받았습니까? | 상담사 | 인원 | 상담사 충원 |
| 3 | CO측정, 혈압, 체중 등을 충분히 체크를 받으셨습니까? | 장비 | 장비 | 장비 확충 |
| 4 | 금연상담사나 다른 직원들이 친절하게 잘 대해 주었습니까? | 상담사 | 서비스 | 상담사 교육 |
| 5 | 국가금연지원서비스를 정기적으로 방문하는 것이 불편하였습니까? | 서비스 | 접근성 | 서비스 다양화 (ex금연버스) |
| 6 | 국가금연지원서비스 이용이 금연성공에 얼마나 도움이 되었습니까? | 삭제 | | |
| 7 | 담배를 피우는 다른 사람에게도 국가금연지원서비스를 이용하도록 권유할 생각이 있습니까? | 종속변수 | | |

<표1> 설문지 내용 및 분류

* 1. **분석 목표**

본 연구에서의 목표는 7번 문항(담배를 피우는 다른 사람들에게도 국가금연지원서비스를 이용하도록 권유할 생각이 있습니까?)을 종속변수로 설정 후 분석을 통해 각 문항의 중요도 순서로 나열하는 것이다.

첫째 목표: 모델링을 통해 7번 문항을 예측  
둘째 목표: 모델링의 결과를 토대로 1~5번 문항의 중요도를 순서대로 나열

* 1. **데이터소개**

설문 데이터는 공공데이터포털에서 제공한다[[1]](#footnote-1). 

<표2> 설문지 데이터

데이터에는 기관유형, 지역, 서비스구분, 제공기관, 출생년도, 성별, 등록유형 및 문항들이 포함된다. 분석목표과 상관없는 기관유형, 지역, 서비스구분, 제공기관, 등록유형 변수는 제거한다. 출생년도는 나이가 적은 순서로 많은 순으로 정수 1에서 8로 대체해주고 연속형 타입으로 설정한다. 성별은 남자가 1 여자가 0으로 명목형 변수로 대체한다. 문항들은 리커트 척도 변수로 0~4까지 다섯개의 값이 있다. 순서형 타입이지만 독립변수로 사용될 때 연속형으로 설정해도 충분한 설명을 갖는다[[2]](#footnote-2). 문항7은 종속변수로 순서형 타입의 데이터이다.

* 1. **분석모델**
* **Ordinal Regression (logit)**
  + 모델

<식1>Ordinal Regression 식

<식1>은 Ordinal Regression식으로서 순서형 종속변수를 회귀모형으로 모델링 할 때 주로 쓰인다. 각 독립변수의 coefficient (**)와** 각 클래스의threshold ()가 통계량으로 추정 및 검정이 가능하다.

* + 독립변수 중요도 분석

각 독립 변수의 coefficient 추정치의 절대값이 해당 변수가 종속변수 결정에 미치는 영향의 크기를 나타낸다. 통계적 추정이기 때문에 신뢰구간을 구해 결과의 신뢰성을 검정할 수 있다.

* **Random Forest**
  + 모델

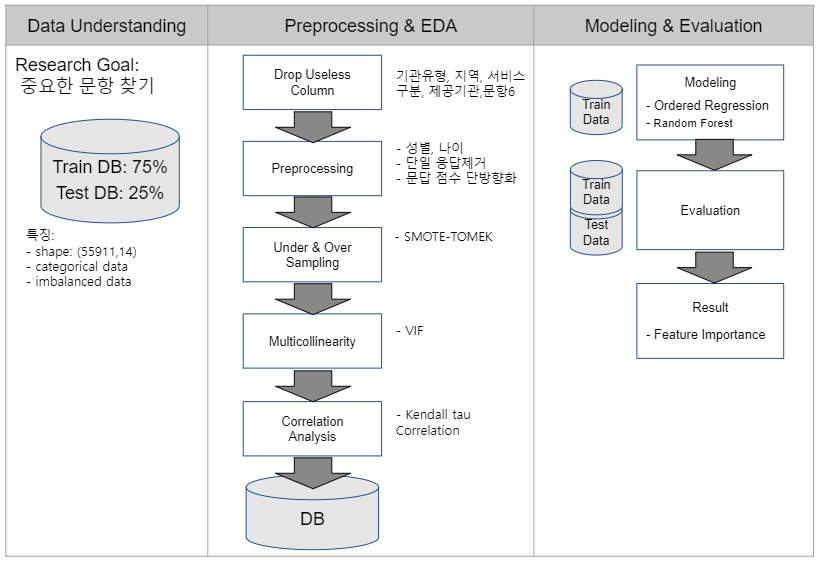
Random Forest는 여러 개의 트리모델들을 앙상블하는 모델이다. 독립변수들 중에 랜덤으로 몇 개만 뽑아 트리모델을 생성하기 때문에 다중공선성을 모델 스스로 다소 해결할 수 있다. 여러 개의 트리모델을 생성하여 Random Forest를 형성한다.

* + 독립변수 중요도 분석

Random Forest는 기본적으로 black-box(모델링의 과정을 자세히 알 수 없는)모델이기 때문에 모델 내에 변수들의 중요도를 대변하는 통계량이 존재하지 않는다. Feature importance 방법은 모델 학습 후 input의 독립변수들 중 하나를 랜덤값으로 바꾸고 예측 및 평가를 해서 어떤 변수가 랜덤이 되었을 때 예측력이 가장 저하하는가를 비교한다.

1. **본론**

**2-1. 분석 플로우**



<표3> 전체 분석 플로우

<표3>은 분석의 전체 플로우를 나타내고 있다. EDA의 결과를 바탕으로 데이터 전처리후 데이터의 불균형을 리샘플링하여 해소한다. 그후 다중공선성 제거와 독립변수의 유의성 검사를 진행한다. Ordinal Regression과 Random Forest모델링으로 독립변수의 중요도를 순서대로 나열한다.

* 1. **EDA 및 전처리**

1. **Drop Useless Columns**

데이터 중에 지역 관련 변수 “지역”과 서비스 분류 관련 변수 “기관유형”, “서비스구분”, “제공기관”, “등록유형”는 분석 목표와 관련이 없기 때문에 제거한다. “문항6”은 개선 방안이 없기 때문에 분석이 불필요한 변수이다.

1. **Preprocessing**

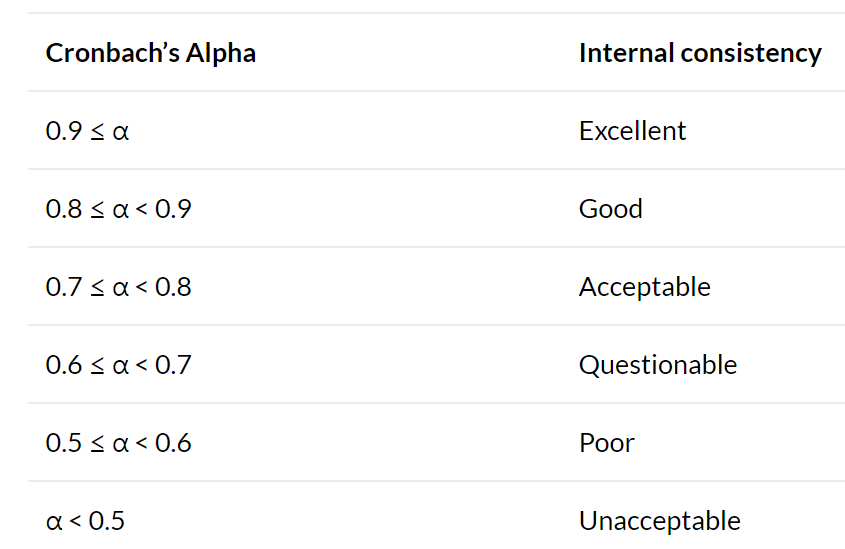
“성별”은 남자는 1 여자는 0인 명목형 데이터로 처리한다.

“나이”는 나이가 적은 순서로 많은 순으로 정수 1에서 8로 대체해주고 연속형 타입으로 설정한다.

문항 1~5까지 단일 응답인 경우 제거한다. 문항5는 다른 문항들과 긍정 부정의 방향이 다르기 때문에 단일 응답인 경우를 제거의 근거가 된다. 또 문항1~5까지 단일 응답인 경우 문항7(종속변수)가 같은 응답인 경우가 99.7%였다.

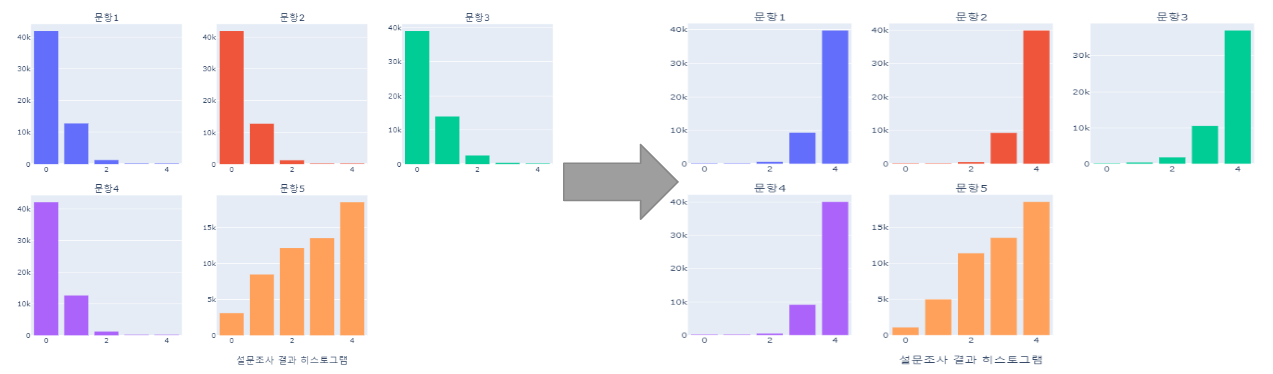
문항 1~ 4번은 0: “매우그렇다”, 1: “그렇다”, 2: “보통이다”, 3: “그렇지 않다”, 4: “전혀 그렇지 않다.” 순서가 정해져 있다. 이를 문항5와 같이 순서를 바꿔 수가 클수록 긍정적 답변이 되게 전처리한다.

전처리 후 설문의 일관성과 신뢰성을 측정하는 크론바흐 알파 테스트 계수 값은 0.772, 신뢰구간(95%)은 [0.769, 0.775] 이다. 크론바흐 알파값은 0과 1사이에 위치하며 1에 가까울수록 서베이의 신뢰성을 보장한다. <표4>는 기본적인 크론바흐 알파 값의 신뢰도 기준이다.[[3]](#footnote-3)



<표4> 크론바흐 알파 값의 신뢰도 기준

* + **전처리 결과**



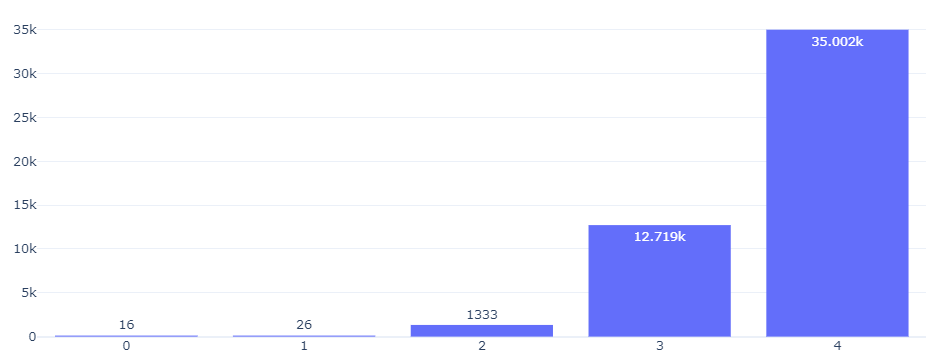
(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<그림1> 전처리전/후 문항1~5 히스토그램 변화

<그림1>은 문항1~5의 전처리 전과 후 히스토그램의 변화를 나타낸다. 단일응답 제거와 응답의 긍정부정 단일화를 거쳤다. 데이터의 불균형이 심한 것을 확인하였고 이를 해결하기 위해 리샘플링을 한다.

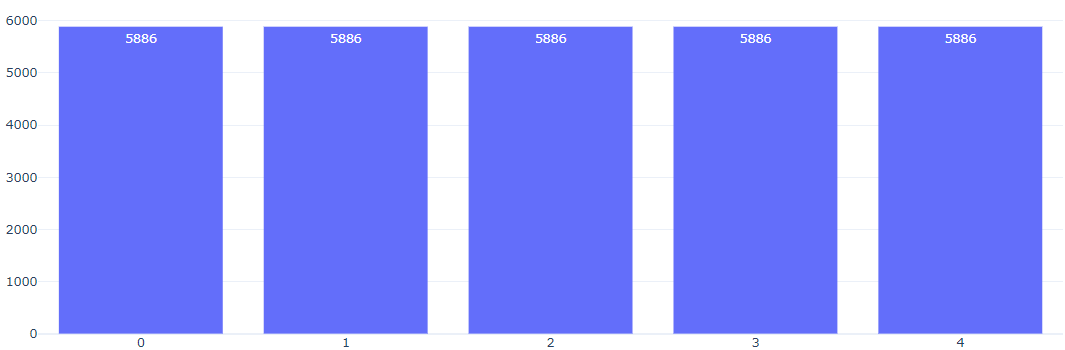
* 1. **Re-sampling**

데이터의 불균형이 심하여 SMOTE-TOMEK기법과 Random Under sampling 기법을 combination 하여 리샘플링으로 데이터의 불균형을 해소한다.



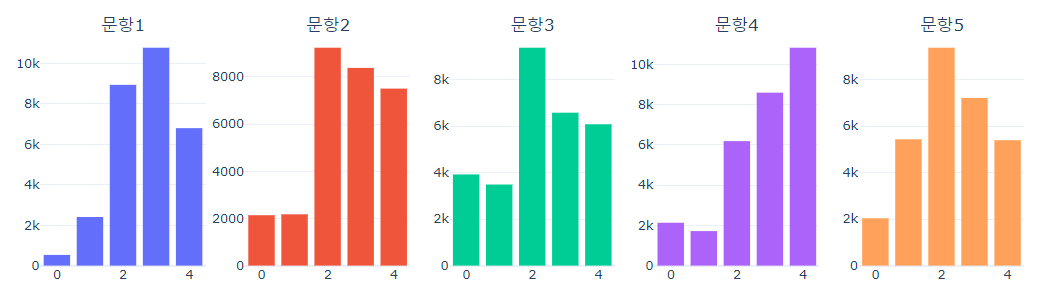
(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<그림2> 리샘플링 전 문항7의 히스토그램



(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<그림3> 리샘플링 후 문항7의 히스토그램



(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

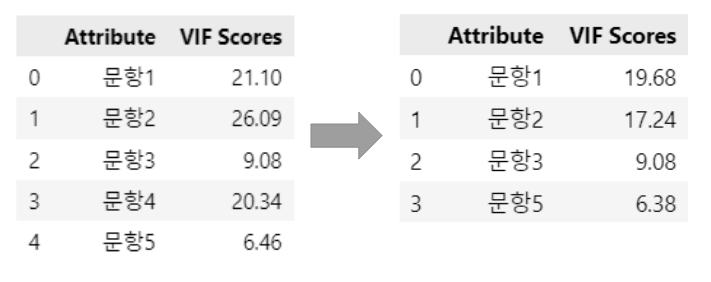
<그림4> 리샘플링 후 문항1~5의 히스토그램

<그림3>와 <그림3>은 리샘플링 전/후 문항7의 히스토그램이다. 샘플링 후 문항7의 각 항목들의 비율이 같아진 것을 볼 수 있다. <그림4>는 리샘플링 후 문항1~5의 히스토그램이다. 불균형이 크게 해소된 것을 볼 수 있다.

리샘플링 후 크론바흐 알파계수는 0.901, 신뢰구간(95%) [0.900, 0.903]으로 전에 비해 크게 상승한 것을 볼 수 있다.

* 1. **Multicollinearity 제거**

다중공선성 VIF(Variance Inflation Factor)를 계산해 여러 시나리오를 확인한 결과 문항4를 제거하는 것이 최선의 방법이다. 문항 1과4는 설문의 분류와 개선방법이 같기 때문에 하나를 제거해도 분석에 지장을 주지 않는다.



<그림5>문항4 제거 전/후 VIF 점수

<그림5>는 문항4 제거 전/후의 VIF 점수를 나타낸다. VIF점수가 10이 넘는 문항이 두개 존재하지만 더 이상 문항을 제거하면 데이터의 설명력이 현저히 저하되어 제거할 수 없다. 다중공선성 이슈를 완전히 해소하지 못한 것이 추후 분석에 계속 영향을 미치게 된다.

* 1. **Correlation Analysis**

Kendall Tau Correlation은 상관분석의 일종으로 두 변수가 선형적 관계를 얼마나 어떻게 맺고 있는가를 수치화 하는 방법이다. 두 연속형 변수 간의 순위를 비교하여 연관성을 계산하는 방법으로 주로 데이터의 샘플 사이즈가 작거나 데이터의 동률이 높을 때 유용한다. 값은 -1일 때 완벽한 반비례, 0일 때 관계없음, 1일 때 완벽한 정비례를 뜻한다.



<표5> kendall tau 점수

<표5>는 문항7과 다른 문항들 간의 kendall tau값을 나타내며 통상 0.8이 넘어야 강한 유의성을 0.6이상은 유의성을 보장한다. 문항5는 0.438이지만 분석목표상 제거하지 않고 분석을 진행한다.

* 1. **Evaluation 방법**
* **Confusion-Matrix**

Confusion Matrix는 명목형 종속변수의 모델의 성능을 가시화하기 좋은 가장 기본적인 방법이다. 종속변수에 5개의 클래스가 존재하기 때문에 매트릭스는 5X5가 되며 아래의 <표6>와 같다.

|  |  |
| --- | --- |
| CONFUSION-MATRIX | PREDICTIVE VALUE |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | 0 | 1 | 2 | 3 | 4 |
| ACTUAL VALUE | 4 |  | FP |  |  | TP |
| 3 |  | FP |  | TP |  |
| 2 |  | FP | TP |  |  |
| 1 | FN | TP | FN | FN | FN |
| 0 | TP | FP |  |  |  |

(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<표6> Confusion Matrix

* **Precision, Recall, Accuracy**

불균형에서 시작된 데이터이기 때문에 Accuracy와 더불어 Recall과 Precision까지 함께 고려하는 것이 현명하다. Precision과 Recall은 각 클래스 별로 산출하여 고려한다. 주의 할 점은 매 클래스 마다 FP와 FN이 달라지는 것이다. 해당 클래스를 기준으로 생각해야한다.

<식1> precision 과 recall 식

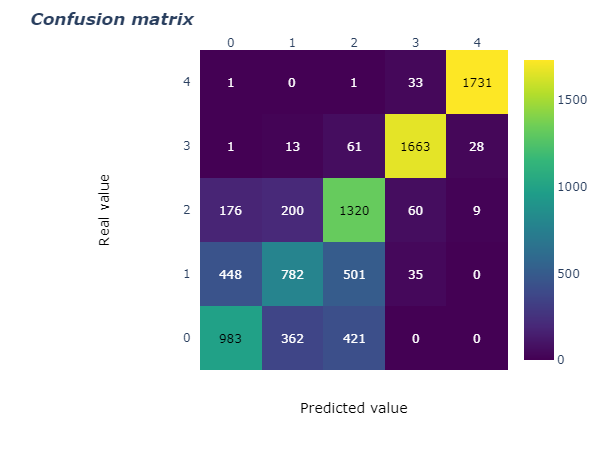
Ex) 클래스1 precision과 recall   
Precision = TP/(TP+FP)

Recall = TP/(TP+FN)

<식1>은 Precision과 Recall의 수식이다.

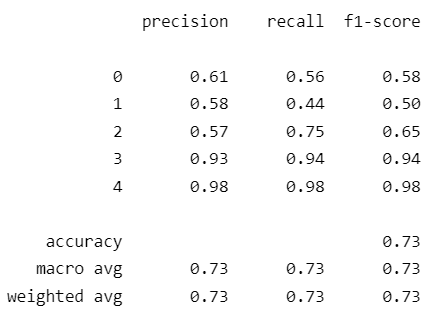
* 1. **분석결과 및 토의**

1. **Ordinal Regression (logit)**



(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<그림6> Ordinal Regression Confusion Matrix

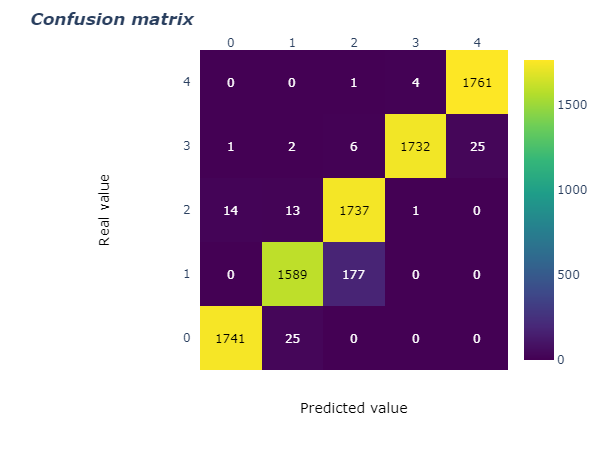


(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<표7> Ordinal Regression Evaluation Table

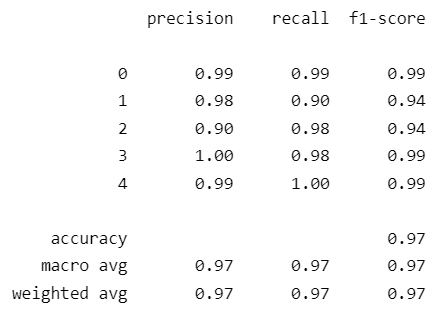
Ordinal Regression 결과는 <그림6>과 <표7>과 같다. <그림6>에서 나타나듯 0~2문항의 예측성능이 떨어진다. 이는 리샘플링 과정에서 나타난 오차를 모델이 잘 분류 못하는 것으로 분석할 수 있다. 또한 precision과 recall 점수 또한 그 클래스들에서 현저히 낮은 것을 볼 수 있다. 성능 저하의 요인으로 다중공선성을 제거하지 못한 점을 들 수 있다. 회귀모델은 다중공선성에 취약함으로 달리 얘기하면 본 모델이 데이터에 적합하지 않다는 결론을 내릴 수 있다.

1. **Random Forest**



(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<그림7>Random Forest Confusion Matrix

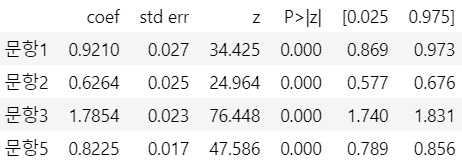


(0: 전혀 그렇지 않다. 1: 그렇다. 2: 보통이다. 3그렇다. 4: 매우 그렇다.)

<표8> Random Forest Evaluation Table

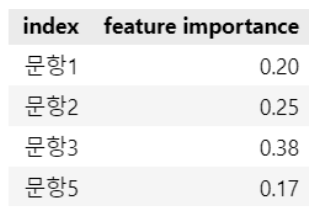
Random Forest 결과는 <그림7>과 <표8>와 같다. Random Forest의 결과는 위와 같이 우수한 것을 볼 수 있다. 다중공선성을 모델 자체적으로 해결할 수 있는 본 모델이 회귀 모델보다 예측 성능이 뛰어나다. 허나 black-box모델이기 때문에 독립변수의 중요도 분석에서 input과 output에 의존한 분석만이 가능한 단점이 있다. 그럼에도 회귀모델에 비해 성능이 현저히 높기 때문에 본 모델을 채택하여 중요도 분석을 진행한다.

1. **Feature Importance**



<표9> Ordinal Regression Feature Importance

<표9>는 Ordinal Regression의 각 독립변수 계수 값이다. 계수 값들은 통계량으로 신뢰구간을 계산해서 유의한지 검증을 할 수 있다. 계수의 절대값이 클수록 해당 독립변수(feature)가 종속변수를 결정하는데 많이 기여한다. 결과를 보면 모든 계수는 신뢰성이 보장되며 변수의 중요도는 “문항3 > 문항1 > 문항5 > 문항2” 이다.



<표10> Random Forest Feature Importance

<표10>는 Random Forest의 feature importance이다. 결과는 “문항3 > 문항2 > 문항1 > 문항5”로 Ordinal Regression의 결과와 다소 상이하다.

1. **결론**

**3-1. 연구의 결론 및 한계**

Ordinal Regression과 Random Forest 모델의 성능의 차이를 고려해서 feature importance를 찾는 방법에 단점이 있음에도 불구하고 Random Forest로 모델을 채택한다. 즉 문항의 중요도는 “문항3 > 문항2 > 문항1(문항4) > 문항5”이다. 그에 따라 서비스 개선을 위해서는 장비 확충에 따른 검사 및 측정 횟수 증가가 가장 시급하다. 또한 상담사의 인원을 확충하여 서비스를 개선해야한다.

연구 중 한계점  
1) 심한 불균형 데이터  
2) 너무 적은 문항

**3-2. 추후 연구**

추후 연구로는 각 항목의 개선비용을 가정하고 시뮬레이션 하여 각 개선방법의 최적 조합을 찾아내야 한다. 또한 과년도의 데이터로 반복 테스트하여 분석의 신뢰성을 높여야 한다. 이에 더불어 모델이 간단하여 직접적 feature importance를 비교할 수 있는 Tree모델을 적용해 보면 더 좋은 분석이 가능 할 것이다.

**REFRENCE**

[1] 한국건강증진개발원\_국가금연지원서비스 등록정보(만족도) <https://www.data.go.kr/data/15092388/fileData.do>

[2] Johnson, D.R., & Creech, J.C. (1983). Ordinal measures in multiple indicator models: A simulation study of categorization error. *American Sociological Review*, 48, 398-407.

[3] Norman, G. (2010). Likert scales, [levels of measurement](https://www.statisticssolutions.com/dissertation-resources/descriptive-statistics/data-levels-of-measurement/) and the “laws” of statistics. *Advances in Health Sciences Education*, *15*(5), pp. 625-632. Retrieved from: <https://link.springer.com/article/10.1007%2Fs10459-010-9222-y#citeas>.

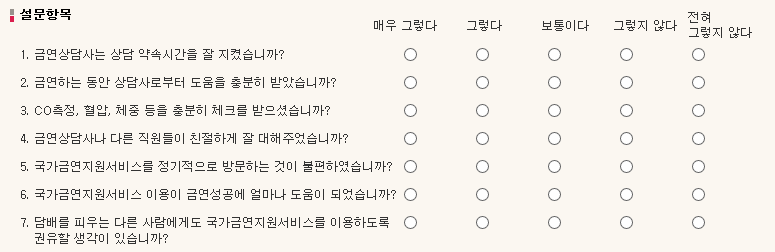
[4] Sullivan, G. & Artino Jr., A. R. (2013). Analyzing and Interpreting Data From Likert-Type Scales. *Journal of Graduate Medical Education*. *5*(4), pp. 541-542

[5] Zumbo, B. D., & Zimmerman, D. W. (1993). Is the selection of statistical methods governed by level of measurement? *Canadian Psychology*, 34, 390-400.

[6] 박현아. (2021). 표본조사에서 크론바흐알파값을 사용한 신뢰성. *응용통계연구*, *34*(1), 1-8.

**APPENDIX A**

실제 설문지



**APPENDIX B**

Source Code

# base tool

import pandas as pd

pd.set\_option('display.max\_columns', 50)

pd.set\_option('display.max\_rows',100)

import numpy as np

from sklearn.model\_selection import train\_test\_split

import copy

import warnings

warnings.filterwarnings('ignore')

#visualization

from IPython.display import display

import matplotlib.pyplot as plt

from matplotlib import font\_manager, rc

font\_path = "C:/Windows/Fonts/gulim.ttc"

font = font\_manager.FontProperties(fname=font\_path).get\_name()

rc('font', family=font)

import plotly.io as pio

import plotly.express as px

import plotly.graph\_objects as go

import plotly.figure\_factory as ff

from plotly.subplots import make\_subplots

pio.templates.default = "plotly\_white"

# test

from imblearn.under\_sampling import RandomUnderSampler

import pingouin as pg

from scipy.stats import chi2\_contingency,shapiro

from statsmodels.stats.outliers\_influence import variance\_inflation\_factor

# resampling

from collections import Counter

from imblearn.combine import SMOTEENN

from imblearn.combine import SMOTETomek

# modeling

from pandas.api.types import CategoricalDtype

from statsmodels.miscmodels.ordinal\_model import OrderedModel

from sklearn.ensemble import RandomForestClassifier

# evaluation

from sklearn.metrics import accuracy\_score,confusion\_matrix, plot\_confusion\_matrix,f1\_score, classification\_report

def summary(df, pred=None):

obs = df.shape[0]

Types = df.dtypes

Counts = df.apply(lambda x: x.count())

Min = df.min()

Max = df.max()

Uniques = df.apply(lambda x: x.unique().shape[0])

Nulls = df.apply(lambda x: x.isnull().sum())

print('Data shape:', df.shape)

if pred is None:

cols = ['Types', 'Counts', 'Uniques', 'Nulls', 'Min', 'Max']

st = pd.concat([Types, Counts, Uniques, Nulls, Min, Max], axis = 1, sort=True)

st.columns = cols

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\nData Types:')

print(st.Types.value\_counts())

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

return st

def vif(df:pd.DataFrame)->None:

print('vif score')

df = df.iloc[:,:-1]

vif\_scores = pd.DataFrame()

vif\_scores["Attribute"] = df.columns

# calculating VIF for each feature

vif\_scores["VIF Scores"] = [round(variance\_inflation\_factor(df.values, i) ,2)for i in range(len(df.columns))]

display(vif\_scores)

print('-'\*50)

def pairwise(df:pd.DataFrame,disp:bool =False)->None:

print('kendall correlation')

print()

df = df.iloc[:,:]

corr = df.corr(method='kendall').round(3)

display(corr.iloc[:-1,[-1]])

# (1,2) (2,4) (4,1)? (6,7)

# 1,2,4 번은 모두 상담사의 관한 질문

if disp:

fig = go.Figure(data=go.Heatmap(

z=corr,

x=corr.columns,

y=corr.columns,

colorscale=px.colors.diverging.RdBu,

zmin=-1,

zmax=1

))

fig.show()

print('-'\*50)

def cronbach (df:pd.DataFrame)->None:

print('cronbach-alpha test:',pg.cronbach\_alpha(data=df,ci=0.95))

print('-'\*50)

def chi2(df:pd.DataFrame)->None:

print('chi-square test')

chi\_df = df.iloc[:,:-1]

y\_ = df.iloc[:,-1]

for i in range(len(chi\_df.columns)):

cross\_tb = pd.crosstab(chi\_df.iloc[:,i],y\_)

chi, p, dof, expected = chi2\_contingency(cross\_tb)

print('문항',str(i+1),':',end=' ')

print(f"chi 스퀘어 값: {round(chi,2)}",

f"p-value (0.05): {p}",

# f"자유도 수: {dof}",

# f"기대값: \n{pd.DataFrame(expected)}",

# f"측정값: \n{cross\_tb}", sep = "\n"

)

print('-'\*50)

def plot\_cm(y\_test,y\_pred):

print('test confusion matrix:')

z = confusion\_matrix(y\_true=y\_test,y\_pred= y\_pred)

x = ['0','1','2','3','4']

y = ['0','1','2','3','4']

# change each element of z to type string for annotations

z\_text = [[str(y) for y in x] for x in z]

# set up figure

fig = ff.create\_annotated\_heatmap(z, x=x, y=y, annotation\_text=z\_text, colorscale='Viridis')

# add title

fig.update\_layout(title\_text='<i><b>Confusion matrix</b></i>',

#xaxis = dict(title='x'),

#yaxis = dict(title='x')

)

# add custom xaxis title

fig.add\_annotation(dict(font=dict(color="black",size=14),

x=0.5,

y=-0.15,

showarrow=False,

text="Predicted value",

xref="paper",

yref="paper"))

# add custom yaxis title

fig.add\_annotation(dict(font=dict(color="black",size=14),

x=-0.35,

y=0.5,

showarrow=False,

text="Real value",

textangle=-90,

xref="paper",

yref="paper"))

# adjust margins to make room for yaxis title

fig.update\_layout(margin=dict(t=50, l=200))

# add colorbar

fig['data'][0]['showscale'] = True

fig.show()

def modeling(df:pd.DataFrame)->None:

## ordinal regression

X = df.iloc[:,:-1]

y = df.iloc[:,-1]

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size = 0.3,random\_state=0,stratify=y)

shapiro\_y = copy.deepcopy(y\_train)

print('count of train y:',sorted(Counter(y\_train).items()))

cat\_type = CategoricalDtype(categories=[0,1,2,3,4], ordered=True)

y\_train = y\_train.astype(cat\_type)

# for distr in ['probit','logit']:

for distr in ['logit']:

print('Ordinal regression',distr)

mod = OrderedModel(y\_train,

X\_train,

distr=distr)

res = mod.fit(method='bfgs')

display(res.summary())

# train data

predicted = res.model.predict(res.params, exog=X\_train)

y\_pred = pd.Series([i.argmax() for i in predicted]).ravel()

if distr =='probit':

residual = shapiro\_y -y\_pred

print(shapiro(residual))

fig = px.histogram(residual)

fig.show()

# test data

predicted = res.model.predict(res.params, exog=X\_test)

y\_pred = pd.Series([i.argmax() for i in predicted]).ravel()

print('-'\*50)

plot\_cm(y\_test=y\_test,y\_pred=y\_pred)

print(classification\_report(y\_test, y\_pred))

print('-'\*50)

print()

## random forest

print('random forest:')

clf = RandomForestClassifier()

clf.fit(X\_train,y\_train)

display(pd.DataFrame({'index':X.columns,'feature importance':clf.feature\_importances\_}).round(2))

y\_pred = clf.predict(X\_test)

print(classification\_report(y\_test, y\_pred))

print()

plot\_cm(y\_test=y\_test,y\_pred=y\_pred)

print('-'\*50)

def smoteEnn(df:pd.DataFrame):

X=df.iloc[:,:-1]

y=df.iloc[:,-1]

smote\_enn = SMOTETomek(random\_state=0)

X\_resampled, y\_resampled = smote\_enn.fit\_resample(X, y)

print(sorted(Counter(y).items()))

print(sorted(Counter(y\_resampled).items()))

resampled\_df = X\_resampled.merge(y\_resampled,left\_index=True,right\_index=True)

return resampled\_df

def pipeline(df:pd.DataFrame) ->None:

cronbach(df)

vif(df)

# chi2(df)

pairwise(df)

modeling(df)

rus = RandomUnderSampler(random\_state=0)

X\_resampled, y\_resampled = rus.fit\_resample(X\_train, y\_train)

sorted(Counter(rawData.iloc[:,-1]).items())

# # Introduction

# # 1. Data Overview

# ## 1.1 data

# https://www.data.go.kr/data/15092388/fileData.do

# 한국건강증진개발원\_국가금연지원서비스 등록정보(만족도)

raw\_data = pd.read\_csv('./만족도평가(2020).csv',encoding='cp949')

rawData = pd.read\_csv('./만족도평가(2020).csv',encoding='cp949')

display(raw\_data.head())

print(raw\_data.shape)

raw\_data = raw\_data.drop(['문항6'],axis=1)

summary(raw\_data)

# # 2.EDA

x = list(raw\_data.columns)

x = x[7:]

fig = make\_subplots(rows=2, cols=3,subplot\_titles=x,vertical\_spacing=0.1,x\_title='설문조사 결과 히스토그램')

trace0 = go.Histogram( x=raw\_data[x[0]],)

trace1 = go.Histogram( x=raw\_data[x[1]],)

trace2 = go.Histogram( x=raw\_data[x[2]],)

trace3 = go.Histogram( x=raw\_data[x[3]],)

trace4 = go.Histogram( x=raw\_data[x[4]],)

trace5 = go.Histogram( x=raw\_data[x[5]],)

fig.append\_trace(trace0,1,1)

fig.append\_trace(trace1, 1, 2)

fig.append\_trace(trace2, 1, 3)

fig.append\_trace(trace3, 2, 1)

fig.append\_trace(trace4, 2, 2)

fig.append\_trace(trace5, 2, 3)

fig.layout.update(height=700)

fig.update\_layout(bargap=0.2)

fig.show()

# # Single Response

# survey data slicing

survey\_data= raw\_data.iloc[:,7:]

# count unique value in row

val\_num = survey\_data.stack().groupby(level=0).apply(lambda x: len(x.unique().tolist()))

survey\_data['unique\_val'] = val\_num

# replace num of unique val to specific unique value

survey\_data['unique\_val']=survey\_data['unique\_val'].map(lambda x: 'multiple\_val' if x !=1 else x)

survey\_data['unique\_val'] = survey\_data.apply(lambda x: 'only '+str(x['문항1']) if x['unique\_val'] ==1 else x,axis=1)['unique\_val']

print(survey\_data['unique\_val'].value\_counts())

px.pie(survey\_data,names='unique\_val')

# H0: 항목1~5 has same value, then 항목7 has same value

# survey data slicing

survey\_data= raw\_data.iloc[:,7:-1]

# count unique value in row

val\_num = survey\_data.stack().groupby(level=0).apply(lambda x: len(x.unique().tolist()))

survey\_data['unique\_val'] = val\_num

temp = survey\_data.merge(raw\_data['문항7'],left\_index=True,right\_index=True)[survey\_data['unique\_val']==1]

pred\_val = temp['문항1'].ravel()

true\_val =temp['문항7'].ravel()

accuracy\_score(true\_val,pred\_val)

#drop

raw\_data = raw\_data[survey\_data['unique\_val']!=1]

# dummy 성별, 출생년도

raw\_data['성별'].replace({'남':1,'여':0},inplace=True)

year\_col = sorted(raw\_data['출생년도'].unique(),reverse=True)

raw\_data['출생년도'].replace(year\_col,range(len(year\_col)),inplace=True)

raw\_data = raw\_data.loc[:,['출생년도','성별','문항1','문항2','문항3','문항4','문항5','문항7']]

#문항 5외 다른 점수 점수 반대로 설정

for col in ['문항1','문항2','문항3','문항4','문항7']:

raw\_data[col] = raw\_data[col].replace([0,1,2,3,4],[4,3,2,1,0])

raw\_data.to\_csv("preprocessed\_data.csv", index=False, encoding="utf-8-sig")

# # 모델링

rawData = pd.read\_csv('preprocessed\_data.csv',encoding='utf-8')

rawData.shape

# full model without re-sampling

pipeline(rawData)

# 출생년도, 성별 제거 model without re-sampling

rawData = rawData.iloc[:,2:]

pipeline(rawData)

# ## TokMEK

rawData = pd.read\_csv('preprocessed\_data.csv',encoding='utf-8')

rawData = rawData.iloc[:,2:]

X=rawData.iloc[:,:-1]

y=rawData.iloc[:,-1]

smote\_enn = SMOTEENN(random\_state=0)

X\_resampled, y\_resampled = smote\_enn.fit\_resample(X, y)

rus = RandomUnderSampler(random\_state=0)

X\_resampled, y\_resampled = rus.fit\_resample(X\_resampled, y\_resampled)

print(sorted(Counter(y).items()))

print(sorted(Counter(y\_resampled).items()))

r\_resampled = X\_resampled.merge(y\_resampled,left\_index=True,right\_index=True)

temp =r\_resampled.iloc[:,:]

x = list(temp.columns)

fig = make\_subplots(rows=1, cols=5,subplot\_titles=x,vertical\_spacing=0.1,x\_title='설문조사 결과 히스토그램')

trace0 = go.Histogram( x=temp[x[0]],)

trace1 = go.Histogram( x=temp[x[1]],)

trace2 = go.Histogram( x=temp[x[2]],)

trace3 = go.Histogram( x=temp[x[3]],)

trace4 = go.Histogram( x=temp[x[4]],)

fig.append\_trace(trace0,1,1)

fig.append\_trace(trace1, 1, 2)

fig.append\_trace(trace2, 1,3)

fig.append\_trace(trace3, 1, 4)

fig.append\_trace(trace4, 1, 5)

fig.layout.update(height=400)

fig.update\_layout(bargap=0.2)

fig.show()

# full model with re-sampling

pipeline(r\_resampled)

# 출생년도, 성별, 항목4 제거 model with re-sampling

pipeline(r\_resampled.iloc[:,[0,1,2,4,5]])

1. [1]한국건강증진개발원\_국가금연지원서비스 등록정보(만족도) <https://www.data.go.kr/data/15092388/fileData.do> [↑](#footnote-ref-1)
2. [2]Johnson & Creech, 1983;[3] Norman, 2010; [4]Sullivan & Artino, 2013; [5]Zumbo & Zimmerman, 1993) [↑](#footnote-ref-2)
3. [6]박현아. (2021). 표본조사에서 크론바흐알파값을 사용한 신뢰성. 응용통계연구, 34(1), 1-8. [↑](#footnote-ref-3)