REPORT

제목 : Final Project

과 목 명 : 빅데이터분석시각화
학 과 : 정보통계학과
제 출 일 : 2023.12.13
학 번 : 2018015030
성 명 : 이광진

1. Data Choose

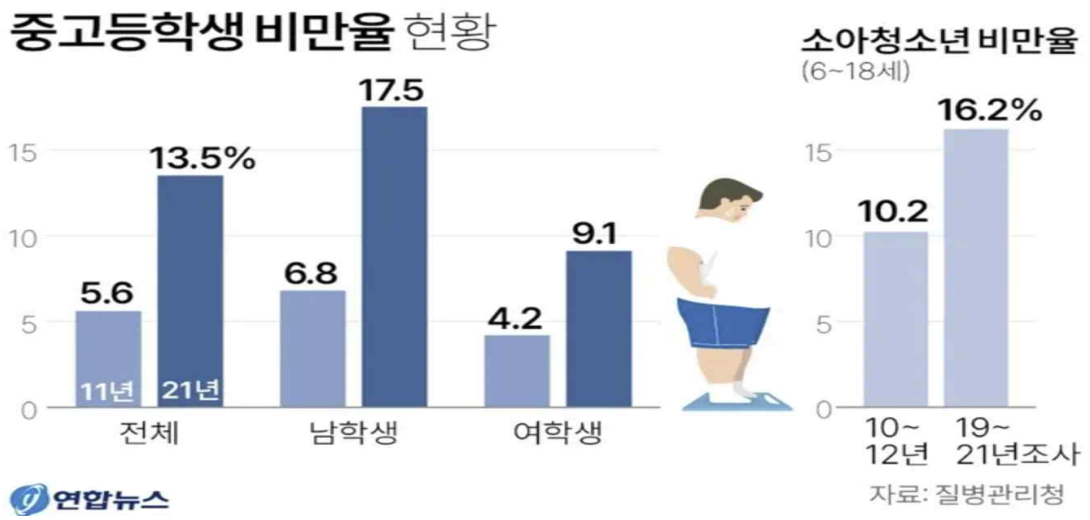
소아·청소년 비만은 성장기의 외모와 건강뿐 아니라 심리와 정서에도 영향을 미쳐 단체 생활에서도 영향을 끼칠 수 있다. 소아·청소년의 비만은 성인 비만으로도 이어질 수 있기에 건강한 개인과 사회를 만들기 위해 청소년기의 비만을 예측하고 예방하는 것이 중요하다. 소아·청소년 비만 현황을 분석하기 위해 학교에서 실시하는 청소년 건강 조사 데이터가 활용될 수 있다. 과거 청소년들의 건강 조사 데이터를 통해 청소년들의 식습관, 생활 습관, 심리, 가정생활 등의 분석을 통해 인사이트를 파악하여 비만의 위험성이 있는 청소년을 구분하는 것을 목적으로 한다. 이를 위해 공공데이터포털의 학생 건강검사 데이터를 불러와 전처리 과정 후 다양한 시각화와 머신러닝 기법들을 이용한다.

[표1] 우리나라 중·고등학생의 연도에 따른 비만율(06~22년) (출처 : KOSIS)

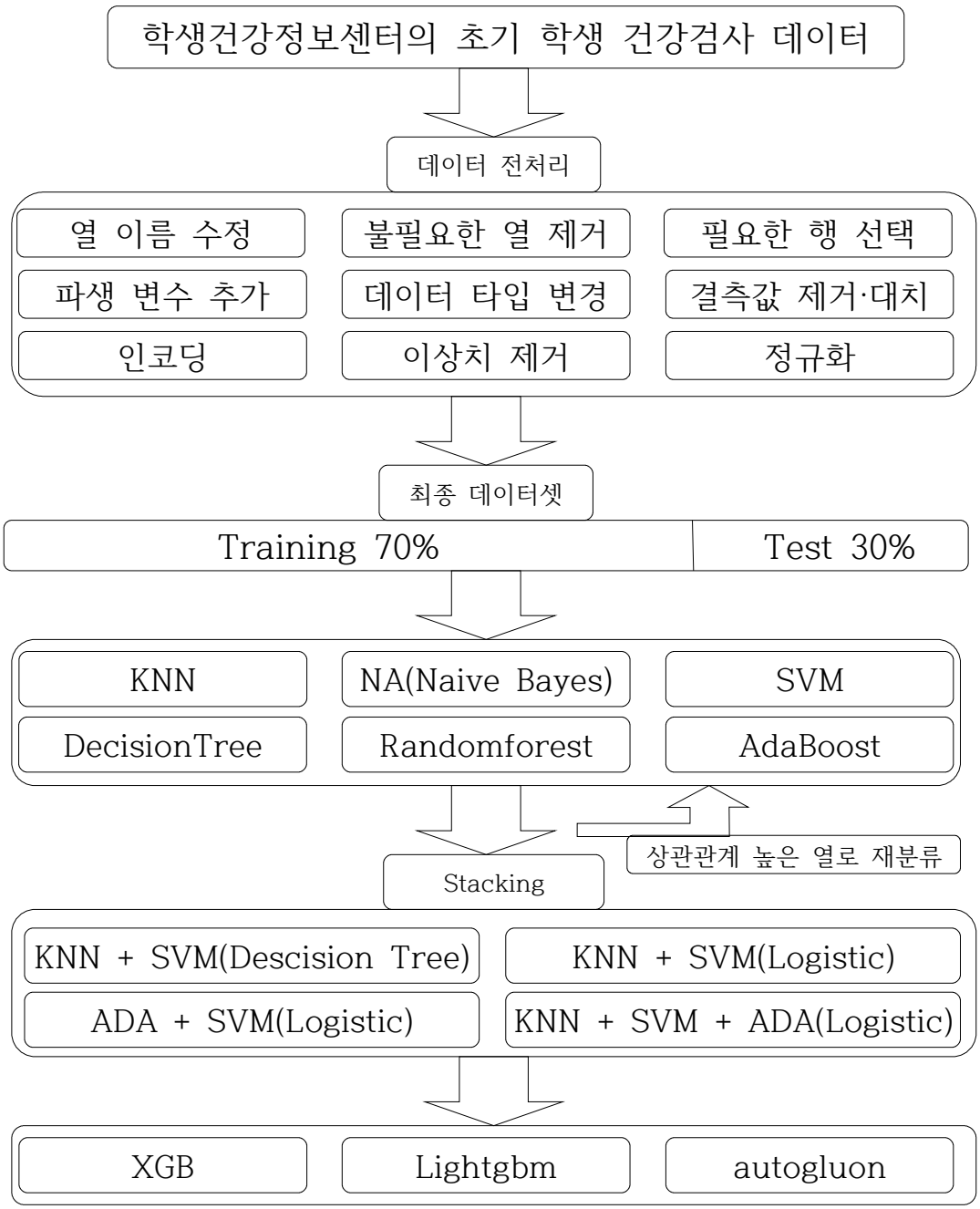
구분	2006	2007	2008	2009	2010	2011	2012	2013
비만율	5.9	5.3	5.3	5.1	5.3	5.6	6.2	6.6

2014	2015	2016	2017	2018	2019	2020	2021	2022
6.9	7.5	9.1	10.0	10.8	11.1	12.1	13.5	12.1

KOSIS를 통해 중·고등학생의 연도에 따른 비만율을 확인한다. 2006년 중·고등학생의 비만율은 전체 학생 수의 5.9%인 반면 2022년 중·고등학생의 비만율은 12.1%로 비만율이 2배 이상 증가한 것을 알 수 있고 2021년 최고치인 13.5% 이후 2022년 약간의 감소를 보이고 있다. [표 1]을 통해 우리나라 중·고등학생의 비만율이 꾸준한 증가세임을 확인할 수 있다. (KOSIS)
[그림1]을 보아 남학생과 여학생의 비만율 증감도 확인할 수 있는데 남학생의 경우 6.8%에서 17.5%로 11년도에 비해 21년도에 약 3배 가까이 증가하였음을 보이고 여학생의 경우 4.2%에서 9.1%로 11년도에 비해 21년도에 약 2배 이상 증가하였음을 보인다.



[그림1] 중고등학생 비만율 현황 (출처 : 경향신문, 연합뉴스 2023, 자료 : 질병관리청)



[그림2] 연구 방법 모형도

본 논문의 연구 방법 모형도는 위의 [그림2]와 같다. 학생건강정보센터의 초기 데이터를 불러와 위와 같은 전처리 과정을 거치고 전처리한 데이터를 이용해 모델을 생성하기 위해 train 데이터와 test 데이터를 7 : 3의 비율로 나눈다. train 데이터로 6가지의 분류 모델을 훈련하고 test 데이터에 적용해 정확도를 확인한다. 그 후 모델 성능 향상을 위해 상관관계 높은 열 추출, Stacking을 실시하고 성능이 좋은 모델을 선택한다.

1. Data Explain

본 논문은 학생건강정보센터 (https://schoolhealth.kr/index.do) 에서 제공하는 설문지 형식의 데이터로 '교육부_학생 건강검사 표본조사 rawdata_20211231'를 이용하여 분석하였다. 해당 데이터를 표현하는 [그림3]은 교육부에서 실시한 학생 건강검사 데이터로 흔히 학교에서 실시하는 건강 검진 데이터이다. 데이터의 열이 113개가 있고 분석에 필요한 데이터들 또한 제한되어 있기 때문에 모든 열들에 대한 소개는 불가능하다. 따라서 열들을 구분하여 소개한다. 학생들의 개인 정보인 [학년, 학교, 도시, 성별, 생년월일] 등이 있으며 건강 검진을 통해 알 수 있는 [키, 몸무게, 시력, 청력, 혈당, 중성지방, 수축기, 이완기, 이상 치아 개수] 등 학생들의 건강 정보에 대한 열을 가지고 있다. 학생들의 건강 정보는 대부분 float타입으로 수치형 변수임을 알 수 있고 해당 열들에 결측값들이 많이 존재하여 이를 정리해 줄 필요가 있다. 학생들의 생활 습관을 알 수 있는 열은 [라면, 음료수, 아침 식사, 다이어트 경험, 운동, 수면량, 씻기, 안전벨트, 게임이용시간] 등이 있고 학생들의 가정과 심리 상태를 알 수 있는 [괴롭힘, 현금갈취, 체별경험, 상담희망, 가족흡연, 가족음주, 무기력감] 등의 열들이 있다. 학생들의 생활습관과 가정, 심리 등을 알 수 있는 열들은 학생들이 직접 설문을 작성하는 항목으로 (1, 2) 또는 (1, 2, 3, 4) 중 하나를 선택하는 명목형 변수로 되어 있다. 해당 열들 또한 결측값들이 존재하여 이를 처리할 필요가 있다. 기본적인 정보(키, 학년, 몸무게 등)를 제외하고는 대부분에서 결측값들(실제로 입력되지 않은 부분도 있지만 초등학생과 중, 고등학생의 설문이 합쳐져 있기에 빠져있는 데이터도 있다.)이 보이는 것을 확인할 수 있고 113개의 열과 97787개의 데이터가 있음을 알 수 있다. int타입이 7개, object타입이 31개, float타입이 75개로 불필요한 열 제거, 분석에 필요한 데이터 선택, 타입변경 등 공공데이터포털에서 제공한 데이터 정보를 제대로 확인하기 위해 데이터 전처리 과정이 필요해 보인다.

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   학년도            97787 non-null  int64
 1   최종가중치          97787 non-null  float64
 2   학교ID            97787 non-null  int64
 3   strata          97787 non-null  int64
 4   도시규모            97787 non-null  object
 5   공학여부            97787 non-null  object
 6   시도              97787 non-null  object
 7   학교급             97787 non-null  object
 8   학년              97787 non-null  int64
 9   반               97787 non-null  int64
 10  순번              97787 non-null  int64
 11  성별              97787 non-null  object
 12  생년월일            97787 non-null  int64
 13  건강검진일           97786 non-null  float64
 14  키               97787 non-null  float64
 15  몸무게             97787 non-null  float64
 16  척추              32230 non-null  object
 17  시력_나안_좌         20826 non-null  float64
 18  시력_나안_우         20826 non-null  float64
 19  시력_교정_좌         11590 non-null  float64
 20  시력_교정_우         11590 non-null  float64
 21  안질환             32492 non-null  object
 22  청력_좌            32441 non-null  object
 23  청력_우            32441 non-null  object
 24  귓병              32492 non-null  object
 25  콧병              32490 non-null  object
 26  목병              32491 non-null  object
 27  피부병             32491 non-null  object
 28  요단백             32350 non-null  object
 29  요잠혈             32338 non-null  object
 30  혈당(식전)(mg_dl)   5084 non-null   float64
 31  총콜레스테롤(mg_dl)   5092 non-null   float64
 32  hdl(mg_dl)      4979 non-null   float64
 33  중성지방(mg_dl)     5025 non-null   float64
 34  ldl(mg_dl)      4955 non-null   float64
 35  AST(U_L)        5074 non-null   float64
 36  ALT(U_L)        5069 non-null   float64
 37  혈색소(g_dl)       5084 non-null   float64
 38  결핵흉부방사선검사       20032 non-null  object
 39  수축기             31969 non-null  float64
```

(중략)

```
 82  안전벨트착용          94013 non-null  float64
 83  안전장비착용          94009 non-null  float64
 84  외상치료경험          56673 non-null  float64
 85  하루tv시청2시간이상     37320 non-null  float64
 86  하루2시간이상게임_초     37350 non-null  float64
 87  하루2시간이상게임_중고    56623 non-null  float64
 88  음란물채팅           56620 non-null  float64
 89  괴롭힘따돌림_초        37332 non-null  float64
 90  괴롭힘따돌림_중고       56644 non-null  float64
 91  현금갈취            37329 non-null  float64
 92  신체접촉            37329 non-null  float64
 93  가출생각_초          37332 non-null  float64
 94  가출생각_중고         56648 non-null  float64
 95  고민상담대상          56646 non-null  float64
 96  가정문제걱정          56647 non-null  float64
 97  가족지지            37346 non-null  float64
 98  체벌경험            37346 non-null  float64
 99  폭력위협            56647 non-null  float64
 100 상담희망_초          37330 non-null  float64
 101 상담희망_중고         56645 non-null  float64
 102 가족흡연            37329 non-null  float64
 103 가족음주            37329 non-null  float64
 104 흡연음주전문가상담희망     56648 non-null  float64
 105 무기력감            37326 non-null  float64
 106 수업태도교정          37347 non-null  float64
 107 과잉행동            9192 non-null   float64
 108 주의력산만           9120 non-null   float64
 109 성문제전문가상담희망      56646 non-null  float64
 110 진로고민            56647 non-null  float64
 111 상담요청_초          37335 non-null  float64
 112 상담요청_중고         56615 non-null  float64
dtypes: float64(75), int64(7), object(31)
memory usage: 84.3+ MB
```

[그림3] '교육부_학생 건강검사 표본조사 rawdata_20211231' 데이터

데이터 설명 코드북에는 120개의 열이 있다고 하지만 실제로는 [호흡기, 비교기, 순환기, ..] 등의 열이 없어 113개의 열이 있음을 확인할 수 있다.

학생들의 건강 상태, 생활 습관, 학교생활, 가정 및 심리 상태가 학생들의 bmi에 어떤 영향을 미치는 지 분석하기를 원하기에 학생들의 키와 몸무게를 통해 bmi라는 새로운 변수를 생성하고 bmi를 [저체중, 정상, 과체중, 비만]으로 나누는 bmi_등급 변수를 생성하여 bmi와 해당 열들의 관계를 분석한다.

해당 데이터의 범주형 변수들은 각 변수에 따라 범주가 설명하는 값들이 다르다. 이를 [표2]로 간략하게 표현하였다.

[표2] 열이 가지는 범주

열 이름 \ 범주	1	2	3	4
라면, 음료수, 패스트푸드, 육류, 우유 유제품, 과일, 채소(김치제외)	먹지 않음.	1-2번	3-5번	매일 먹음.
아침식사	거의 꼭 먹음.	대체로 먹음.	대체로 안 먹음.	거의 안 먹음.
다이어트경험_답변1~4	아무것도 안함.	식단을 조절한다.	약을 먹는다.	운동을 한다.
주3회이상운동	거의 안 했음,	1-2일 정도.	3-4일 정도,	5일 이상
하루수면량	6시간 이내,	6-7시간,	7-8시간,	8시간 이상
하루tv시청2시간이상, 하루2시간이상게임_초, 괴롭힘따돌림_초, 현금갈취, 신체접촉, 가출생각_초, 가족지지, 체벌경험, 학교문제상담희망_초, 가족흡연, 가족음주, 무기력감, 수업태도교정, 상담요청_초	예	아니오		

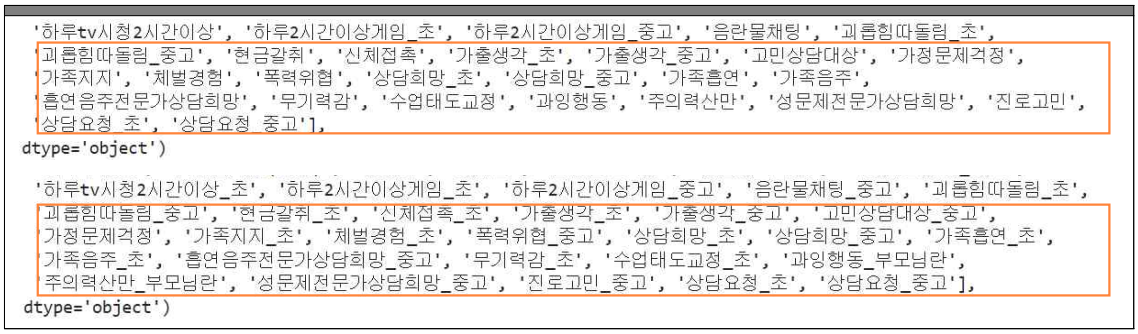
2. Data Processing

분석에 앞서 데이터의 rename, 불필요한 열 제거, 결측값 제거, 이상치 처리 등 데이터 전처리를 하였다.

(1) 열 이름 수정 (Rename)

해당 데이터는 97,787명의 학생들을 대상으로 하여 건강검사를 실시한 데이터로 초등학생부터 고등학생까지의 데이터가 존재한다. 키, 학교, 몸무게 등 기본적인 정보를 설명하는 열은 모든 학생들의 데이터를 가지고 있지만 열들에 따라 약 3만 개의 데이터가 초등학생을, 6만개의 데이터가 중, 고등학생을 설명하는 열로 구분되어 있기 때문에 초등학생 데이터만을 활용하기 위해 중, 고등학생 데이터를 제외하기 위한 작업을 한다. [그림4]에서 열들의 이름이 초

등학생과 중, 고등학생이 구분된 열들이 있는 반면 구분이 안된 열들이 존재하기에 이러한 열들을 구분해 준다. 열 이름에 _초, _중고로 구분되어 있는 열들 이외에 구분되지 않은 열들의 이름을 rename 하였다.

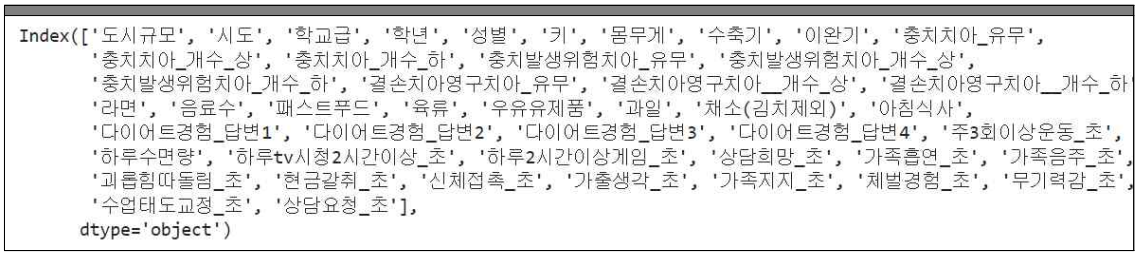


[그림4] 데이터 열 이름 변경 전, 변경 후

이름이 잘 수정되었음을 확인한다.

(2) 불필요한 열 제거(Drop)

bmi에 대해 분석을 할 예정이기에 bmi와 관련이 없다고 생각되는 불필요한 열들을 제거하는 작업이 필요하다. 총 113개의 열이 존재하여 health 데이터의 불필요한 열을 제외하기 위해 drop()을 사용할 수 있지만 해당 데이터는 열들이 너무 많고 하나씩 확인하기 어렵기에 health 데이터의 columns를 가져와 불필요한 열들을 #을 통해 열들을 하나씩 생략하여 health_drop으로 새롭게 정의하였다. [_중고] 가 붙은 데이터들은 중, 고등학생을 의미하는 데이터로 초등학생들의 bmi를 분석하려는 해당 프로젝트에서는 필요가 없는 열들이기에 모두 제거해 주었다. [혈당, 총콜레스테롤, 중성지방,...] 등의 데이터들은 bmi와 연관이 있어 보이지만 결측값이 너무 많아 분석에 적용하기 어려울 것으로 판단하여 제거하였다. 시력, 청력, 귓병, 콧병의 경우 bmi와 연관이 없다고 판단되어 제거하였다. 이상 치아는 bmi와 연관이 없다고도 생각되지만 비만은 주로 에너지 소비량 대비 과도한 음식물 섭취로 발생하여 음식물 섭취와 치아는 큰 연관이 있기에 식습관과 관련이 있다고 판단하여 열을 제거하지 않았다. 그 외에는 주관적인 판단으로 bmi와 연관이 없다고 생각하여 제거하였다. [그림5]는 불필요한 열 제거 후 {도시규모 ~ 상담요청_초}의 46개 열이 남은 것을 보인다.



[그림5] 불필요한 열 제거 후 데이터

(3) 필요한 행 선택(Choose)

해당 데이터는 학교급이 [‘초’, ‘중’, ‘고’]로 구분되어 있다. 초등학생의 비만 데이터 분석을 위해 ‘학교급’이 ‘초’인 행들을 추출한다. [그림6]을 통해 총 97,787개의 데이터 중 38,448개의 데이터가 추출된 것이 보인다.

```
0        초
1        초
2        초
3        초
4        초
        ..
38443    초
38444    초
38445    초
38446    초
38447    초
Name: 학교급, Length: 38448, dtype: object
```

[그림6] 필요한 행 선택 후 데이터

(4) 파생 변수 추가

[그림7]은 파생 변수를 추가한 것으로 bmi는 몸무게/$(\frac{키}{100})^2$이므로 기존 데이터의 키와 몸무게 변수를 이용해 학생에 따른 파생 변수 bmi를 생성한다. [그림7]의 왼쪽 그림을 통해 38,448명의 학생들에 따라 키와 몸무게마다 bmi가 생성된 것을 볼 수 있다. 본 논문에서는 학생들의 bmi의 예측이 아닌 bmi를 이용해 학생들을 비만, 과체중, 정상, 저체중으로 분류하여 등급을 예측하는 분석을 실시할 것이기 때문에 생성된 bmi에 따른 bmi_등급 파생 변수를 생성한다. bmi_등급은 ‘저체중’, ‘정상’, ‘과체중’, ‘비만’으로 나누며 저체중, 정상, 과체중, 비만은 나이와 성별에 따라 bmi가 가지는 값이 다르므로 나이와 성별에 따른 bmi 등급의 기준을 다르게 하여 구분한 것을 [표3], [표4]로 표현하였다.

```
0        16.528926                          0        정상
1        17.888637                          1        정상
2        18.269083                          2        정상
3        15.802112                          3        정상
4        16.148774                          4        정상
           ...                                      ...
38443    23.808022                          38443    과체중
38444    17.426325                          38444    정상
38445    20.692479                          38445    정상
38446    21.316255                          38446    정상
38447    18.569532                          38447    정상
Name: bmi, Length: 38448, dtype: float64    Name: bmi_등급, Length: 38448, dtype: object
```

[그림7] 기존 변수로 추가된 파생 변수

[표3]과 [표4]는 남학생과 여학생의 bmi에 따른 bmi 등급을 표현한 표이다. 남학생, 여학생 모두에서 학년이 증가할수록 bmi 등급이 가지는 bmi의 수치가 증가하는 것을 볼 수 있고 동일 학년의 경우 남학생의 bmi 등급이 가지는 bmi의 수치가 여학생의 bmi 등급이 가지는 bmi 수치보다 더 높은 것을 알 수 있다. [표3], [표4]를 기준으로 [그림7]의 오른쪽 그림과 같이 bmi_등급 변수를 생성하였고 각 학생의 bmi에 따른 bmi_등급을 추가하였다.

[표3] 남학생의 bmi에 따른 bmi 등급

남자 BMI	저체중	정상	과체중	비만
1학년	BMI < 13.93	13.93 ≤ BMI < 18.86	18.86 ≤ BMI < 20.93	20.93 ≤ BMI
2학년	BMI < 14.06	14.06 ≤ BMI < 19.80	19.80 ≤ BMI < 22.13	22.13 ≤ BMI
3학년	BMI < 14.27	14.27 ≤ BMI < 20.76	20.76 ≤ BMI < 23.34	23.34 ≤ BMI
4학년	BMI < 14.57	14.57 ≤ BMI < 21.71	21.71 ≤ BMI < 24.48	24.48 ≤ BMI
5학년	BMI < 14.93	14.93 ≤ BMI < 22.57	22.57 ≤ BMI < 25.5	25.5 ≤ BMI
6학년	BMI < 15.35	15.35 ≤ BMI < 23.32	23.32 ≤ BMI < 26.35	26.35 ≤ BMI

[표4] 여학생의 bmi에 따른 bmi 등급

여자 BMI	저체중	정상	과체중	비만
1학년	BMI < 13.63	13.63 ≤ BMI < 18.27	18.27 ≤ BMI < 20.05	20.05 ≤ BMI
2학년	BMI < 13.77	13.77 ≤ BMI < 19.05	19.05 ≤ BMI < 21.05	21.05 ≤ BMI
3학년	BMI < 14.01	14.01 ≤ BMI < 19.88	19.88 ≤ BMI < 22.09	22.09 ≤ BMI
4학년	BMI < 14.33	14.33 ≤ BMI < 20.71	20.71 ≤ BMI < 23.08	23.08 ≤ BMI
5학년	BMI < 14.73	14.73 ≤ BMI < 21.51	21.51 ≤ BMI < 23.99	23.99 ≤ BMI
6학년	BMI < 15.20	15.20 ≤ BMI < 22.22	22.22 ≤ BMI < 24.77	24.77 ≤ BMI

(5) 수치 데이터(describe)

[그림8]은 describe()로 수치형 데이터를 보이며 이를 통해 각 열들의 개수, 평균, 표준편차, 최소, 최대, 제1사분위-제3사분위의 통계 값을 확인한다. 학년은 1학년부터 6학년까지 있음을 확인할 수 있고 평균이 3.5인 것을 보아 고학년이 조금 더 많음을 확인할 수 있다. 키의 평균은 13cm이고 키가 가장 작은 학생은 102cm, 가장 큰 학생은 180cm로 약 80cm의 큰 차이를 보이는 것을 알 수 있다. 몸무게의 평균은 37kg이고 몸무게의 최소가 14.6kg, 최대가 110kg으로 약 100kg의 차이를 보이며 키보다 몸무게의 차이가 더 큰 것을 확인할 수 있다. 수축기의 경우 평균이 97로 정상 범주를 잘 표현하고 있지만 최댓값이 914, 최솟값이 11로 이상치가 확인되는 것을 알 수 있다. 이는 기계 오류 또는 작성 오류로 판단되므로 데이터 전처리 과정에서 처리 해주어야 한다.

이완기 또한 최댓값이 663, 최솟값이 6으로 이상치를 가지고 있다. 따라서 데이터 전처리 과정에서 처리 해줘야 한다. 이상이 있는 치아 상_하 개수 들의 평균을 보면 모두 0.6개 미만임을 확인할 수 있다. 반면 최댓값은 11, 9, 8로 높은 값을 보이는데 이는 대부분의 학생들이 이상이 있는 치아가 없으나 소수의 학생들에게서 이상 있는 치아가 많이 확인되는 것을 알 수 있다. bmi의 평균은 약 19이고 bmi의 최댓값이 38.8, 최솟값이 10.4로 약 4배 가까운 차이를 보이고 있는 것을 확인할 수 있다. 키, 몸무게와 수축기, 이완기의 값을 보면 다른 열들의 값보다 매우 큰 값을 가지는 것을 보인다. 키와 몸무게는 파생 변수 bmi를 생성 후 분석에서 제거할 예정으로 정규화가 필요하지 않지만 수축기, 이완기의 경우 정규화가 필요해 보인다. 또한 학년, 키, 몸무게 열들의 데이터 개수를 보면 38,448인 반면 이외의 열들은 데이터의 개수가 일정하지 않아 결측값들이 많이 존재하는 것을 알 수 있다. 이들 결측값 또한 제거한다. [가족음주, 괴롭힘, 현금갈취, … , 상담요청]의 경우 평균값이 약 1.9로 대부분 2의 값을 가지는 것을 보아 대부분의 청소년들이 [가족음주, 괴롭힘, 현금갈취, … , 상담요청]에 문제가 없고 도움이 필요하지 않은 것을 알 수 있어 bmi 등급에 영향을 끼치지 못할 것으로 보인다.

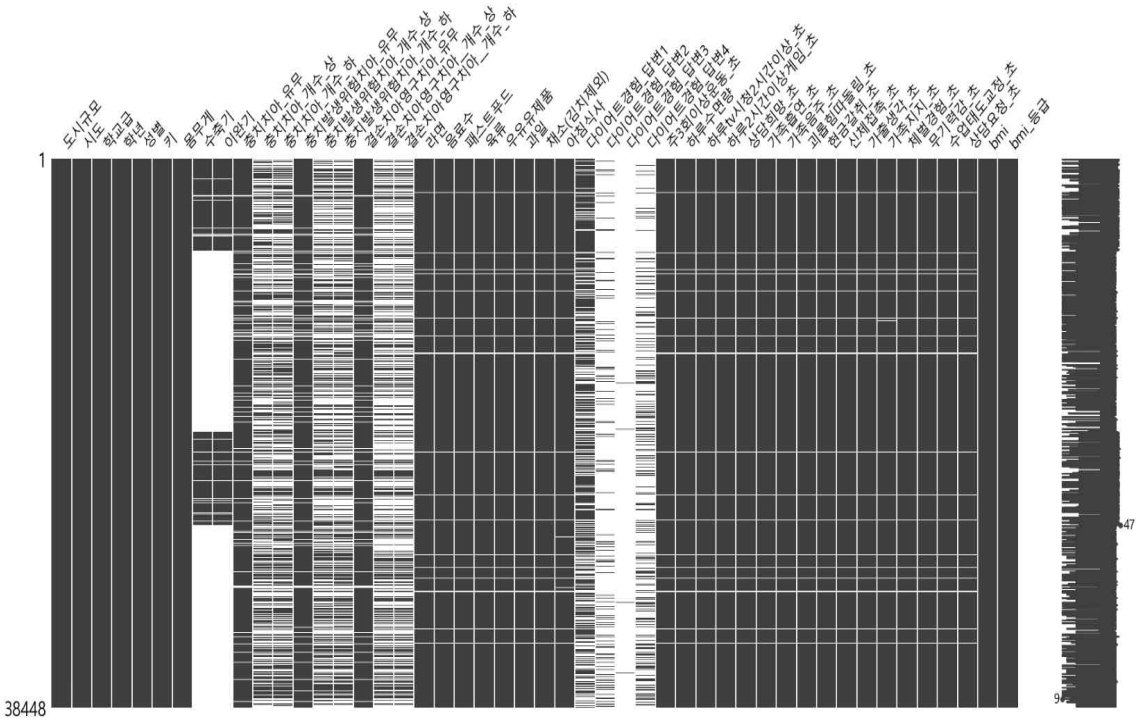
	가족음주_초	괴롭힘따돌림_초	현금갈취_초	신체접촉_초	가출생각_초	가족지지_초	체벌경험_초	무기력감_초	수업태도교정_초	상담요청_초	bmi
	37329.000000	37332.000000	37329.000000	37329.000000	37332.000000	37346.000000	37346.000000	37326.000000	37347.000000	37335.000000	38448.000000
	1.857885	1.968311	1.996196	1.987811	1.950230	1.104241	1.971965	1.961823	1.964522	1.973189	19.058053
	0.349172	0.175172	0.061560	0.109730	0.217472	0.305578	0.165075	0.191626	0.184987	0.161534	3.858631
	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	10.455774
	2.000000	2.000000	2.000000	2.000000	2.000000	1.000000	2.000000	2.000000	2.000000	2.000000	16.083750
	2.000000	2.000000	2.000000	2.000000	2.000000	1.000000	2.000000	2.000000	2.000000	2.000000	18.290888
	2.000000	2.000000	2.000000	2.000000	2.000000	1.000000	2.000000	2.000000	2.000000	2.000000	21.338249
	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	38.806522

	학년	키	몸무게	수축기	이완기	충치치아_개수_상	충치치아_개수_하	충치발생위험치아_개수_상	충치발생위험치아_개수_하	결손치아영구치아__개수_상	결손치아영구치아__개수_하
count	38448.000000	38448.000000	38448.000000	12105.000000	12105.000000	18568.000000	18399.000000	17365.000000	17417.000000	14720.000000	14775.000000
mean	3.497321	137.735079	37.094083	97.099314	59.295679	0.537268	0.498179	0.350993	0.368663	0.010870	0.020846
std	1.703667	12.275926	12.476537	13.288884	10.331656	1.067804	0.993448	0.744623	0.753716	0.133484	0.185719
min	1.000000	101.900000	14.600000	11.000000	6.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2.000000	128.000000	27.400000	90.000000	55.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	4.000000	137.100010	34.799999	98.000000	60.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	5.000000	147.000000	44.400002	104.000000	64.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000
max	6.000000	179.800000	110.600000	914.000000	663.000000	11.000000	9.000000	8.000000	8.000000	4.000000	5.000000

[그림8] 수치화된 데이터

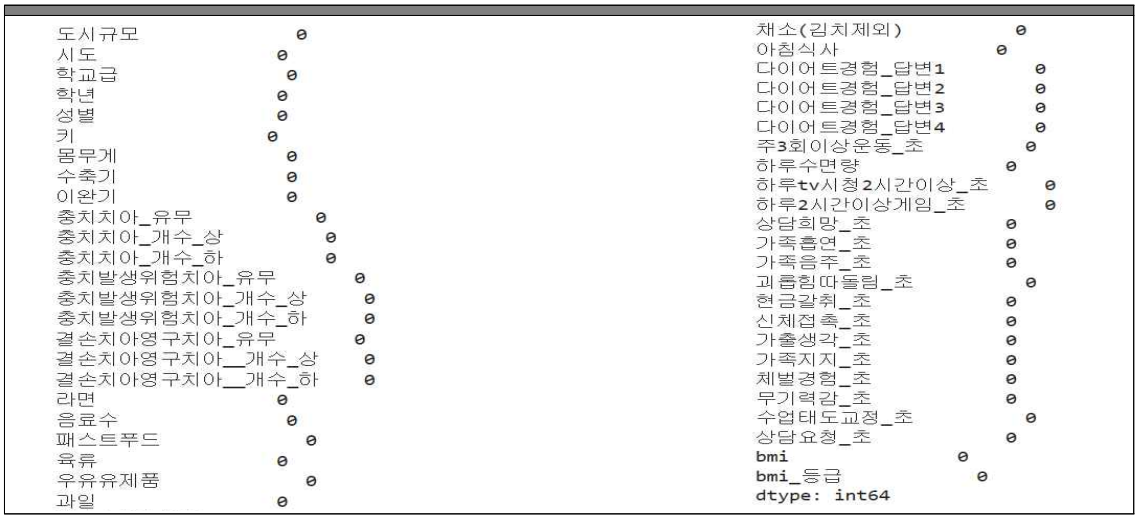
(6) 데이터 타입 변경, 결측값 제거 및 대치

[그림]을 통해 해당 데이터의 결측값을 시각화하였다. 도시규모, 시도, 학교급, 학년, 성별, 키, 몸무게의 경우 모든 학생들의 데이터가 입력되어 결측값이 없는 것을 알 수 있다. 반면 수축기부터 상담요청_초까지의 열들은 결측값들이 존재하는 것을 알 수 있으며 특히 수축기, 이완기, 충치치아_개수_상·하, 충치발생위험치아_개수_상·하, 결손치아영구치아_개수_상·하, 다이어트경험_답변1~4의 경우 결측값들이 많이 존재하는 것을 보인다. 데이터에 문제가 많기에 데이터 전처리를 수행하지 않고 분석을 하는 것은 불가능하다. bmi_등급에 연관이 있는 열들을 기준으로 결측값을 제거 또는 대치하였다.



[그림] 결측값 시각화

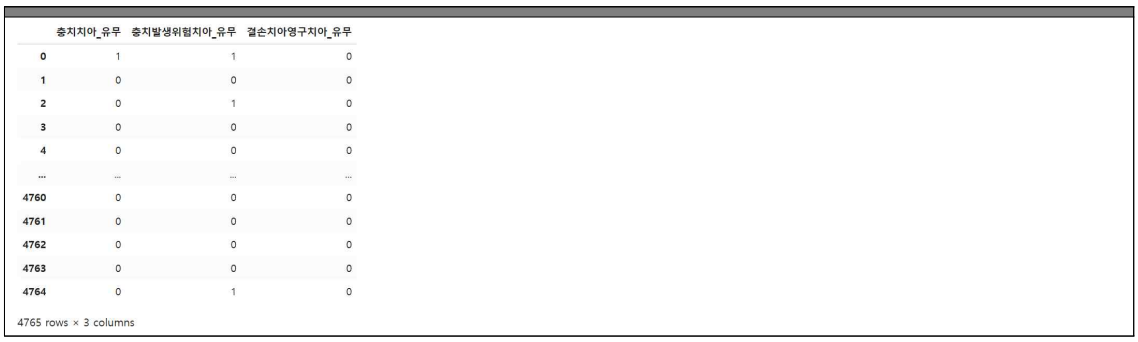
해당 건강검사 데이터는 설문 조사 형식의 데이터로 '라면, ..., 상담요청_초' 까지 (1, 2) 또는 (1, 2, 3, 4)와 같이 항목을 선택하는 범주형 변수이다. 범주형 변수와 수치형 변수에 대해 결측값들을 제거한 후 데이터 타입을 float에서 int로 변환한다. 다이어트경험_답변의 경우 다이어트 경험의 더미 변수로 [답변1 : 하지 않음], [답변2 : 식단을 조절한다.], [답변3 : 약을 먹는다.], [답변4: 운동을 한다.]로 구성되어 답변 1에 해당하면 1, 그렇지 않으면 0 답변 2에 해당하면 2, 그렇지 않으면 0 답변 3에 해당하면 1, 그렇지 않으면 0 답변 4에 해당하면 4, 그렇지 않으면 0으로 표현되어 있는데 각각의 답변에 해당하면 1, 아니면 0으로 처리한다. 데이터의 결측값들이 잘 제거되었는 지 확인한다. [그림9]에서 isnull().sum() 함수를 통해 해당 데이터에 결측값이 없는 것을 알 수 있다.



[그림9] 결측값 확인

(7) 인코딩(encoding)

분석의 편의를 위해 충치 치아, 충치 발생 위험 치아, 결손 치아 영구 치아 유무에 따라 무: 0, 유:1로 부호화하였다. [그림10]에서 인코딩되었음을 확인한다.



	충치치아_유무	충치발생위험치아_유무	결손치아영구치아_유무
0	1	1	0
1	0	0	0
2	0	1	0
3	0	0	0
4	0	0	0
...	...	...	...
4760	0	0	0
4761	0	0	0
4762	0	0	0
4763	0	0	0
4764	0	1	0

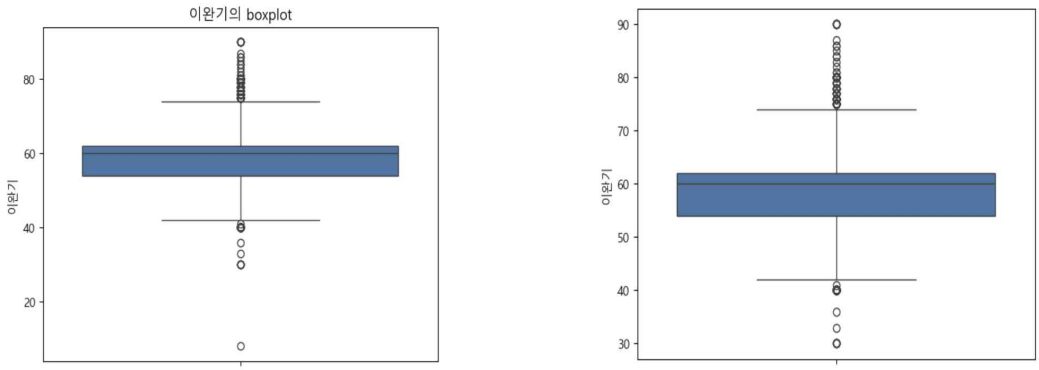
4765 rows × 3 columns

[그림10] 인코딩 데이터

(8) 이상치 제거

이완기의 boxplot을 그려보았다. [그림11]의 왼쪽 그림을 보면 다수의 이상치가 발견되고 그 중 20 미만의 이상치가 발견된다. 20 미만의 이상치를 제외한 다른 이상치들의 경우 실제로 저혈압이나 고혈압의 경우 이완기의 값이 30~40 또는 80 이상의 값을 가질 수 있으므로 이상치를 제거하기엔 무리가 있다. 반면 저혈압의 경우라고 20 미만의 수치는 오류라고 판단되고 분석에 영향을 끼칠 수 있으므로 20 미만의 이상치를 제거한다.

이상치를 제거한 후 boxplot을 그려본다. [그림11]의 오른쪽 그림을 통해 20 미만의 이상치가 잘 제거되었음을 확인한다.



[그림11] 이상치 데이터 제거 전, 제거 후

(9) Min-max 정규화

결측값들과 이상치들을 모두 제거한 후 위에서 확인한 것처럼 다른 열들의 값에 비해 수치가 큰 이완기와 수축기를 Min-max 정규화를 실시한다. [그림12]를 통해 최솟값 60, 최댓값 142, 평균값 96.7을 가지던 수축기는 정규화를 통해 최솟값 0, 최댓값 1, 평균값 0.45를 가지고 최솟값 30, 최댓값 90, 평균값 58을 가지던 이완기는 정규화를 통해 최솟값 0, 최댓값 1, 평균값 0.47을 가지며 정규화가 잘 되었음을 확인한다.

수축기	이완기	충치치아_유무	충치치아_개수_상	충치치아_개수_하	충치발생위험치아_유무
4765.000000	4765.000000	4765.000000	4765.000000	4765.000000	4765.000000
0.447698	0.473204	0.219098	0.280797	0.229171	0.166422
0.124043	0.118989	0.413678	0.794911	0.685055	0.372498
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
0.365854	0.400000	0.000000	0.000000	0.000000	0.000000
0.439024	0.500000	0.000000	0.000000	0.000000	0.000000
0.512195	0.533333	0.000000	0.000000	0.000000	0.000000
1.000000	1.000000	1.000000	7.000000	6.000000	1.000000

수축기	이완기	충치치아_유무	충치치아_개수_상	충치치아_개수_하	충치발생위험치아_유무
4765.000000	4765.000000	4765.000000	4765.000000	4765.000000	4765.000000
96.711228	58.392235	0.219098	0.280797	0.229171	0.166422
10.171562	7.139340	0.413678	0.794911	0.685055	0.372498
60.000000	30.000000	0.000000	0.000000	0.000000	0.000000
90.000000	54.000000	0.000000	0.000000	0.000000	0.000000
96.000000	60.000000	0.000000	0.000000	0.000000	0.000000
102.000000	62.000000	0.000000	0.000000	0.000000	0.000000
142.000000	90.000000	1.000000	7.000000	6.000000	1.000000

[그림12] 데이터 정규화 전, 후

(10) 데이터 전처리 후 확인

[그림13]을 보며 데이터 전처리 전과 후의 데이터를 비교한다. 전처리 전의 경우 모든 학생이 가지는 학년도의 열을 통해 총 97,787개의 데이터가 있고 열들에 따라 결측값들이 존재하는 것을 확인할 수 있지만 초등학생 데이터를 추출하고 결측값들을 제거하여 총 4,766개의 데이터가 남은 것을 보인다.

<데이터 전처리 전>

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 97787 entries, 0 to 97786
Data columns (total 113 columns):
 #    Column          Non-Null Count  Dtype
---   ------          --------------  -----
 0    학년도            97787 non-null  int64
 1    최종가중치          97787 non-null  float64
 2    학교ID           97787 non-null  int64
 3    strata          97787 non-null  int64
 4    도시규모            97787 non-null  object
 5    공학여부            97787 non-null  object
 6    시도              97787 non-null  object
 7    학교급             97787 non-null  object
 8    학년              97787 non-null  int64
 9    반               97787 non-null  int64
 10   순번              97787 non-null  int64
 11   성별              97787 non-null  object
 12   생년월일            97787 non-null  int64
 13   건강검진일           97786 non-null  float64
 14   키               97787 non-null  float64
 15   몸무게             97787 non-null  float64
 16   척추              32230 non-null  object
 17   시력_나안_좌         20826 non-null  float64
 18   시력_나안_우         20826 non-null  float64
 19   시력_교정_좌         11590 non-null  float64
 20   시력_교정_우         11590 non-null  float64
(생략)
```

<데이터 전처리 후>

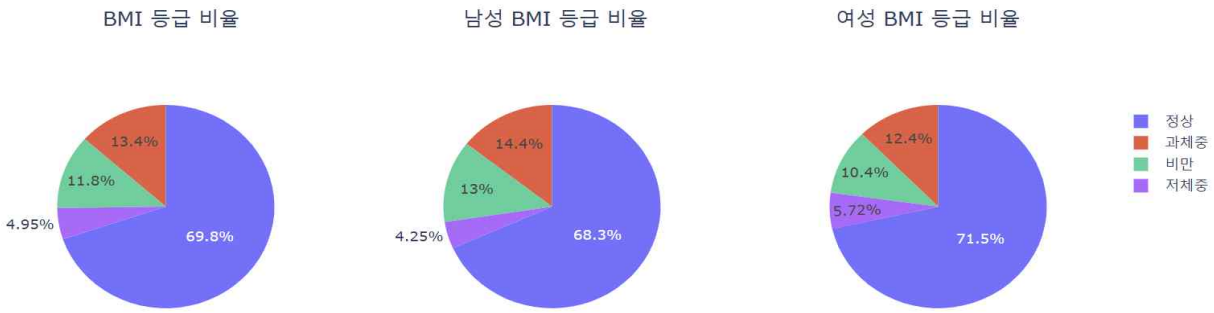
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4766 entries, 12 to 25686
Data columns (total 48 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   도시규모            4766 non-null   object
 1   시도              4766 non-null   object
 2   학교급             4766 non-null   object
 3   학년              4766 non-null   int64
 4   성별              4766 non-null   object
 5   키               4766 non-null   float64
 6   몸무게             4766 non-null   float64
 7   수축기             4766 non-null   float64
 8   이완기             4766 non-null   float64
 9   충치치아_유무          4766 non-null   object
 10  충치치아_개수_상        4766 non-null   int32
 11  충치치아_개수_하        4766 non-null   int32
 12  충치발생위험치아_유무      4766 non-null   object
 13  충치발생위험치아_개수_상    4766 non-null   int32
 14  충치발생위험치아_개수_하    4766 non-null   int32
 15  결손치아영구치아_유무      4766 non-null   object
 16  결손치아영구치아__개수_상  4766 non-null   int32
 17  결손치아영구치아__개수_하  4766 non-null   int32
 18  라면              4766 non-null   int32
 19  음료수             4766 non-null   int32
 20  패스트푸드           4766 non-null   int32
(생략)
```

[그림13] 데이터 전처리 전, 후

3. Data Visualization

(1) Pie Chart

[그림14]는 bmi 비율을 표현한 것으로 전체 성별, 남성, 여성에 따라 bmi_등급의 비율을 각각 표현하였다. 세 부분에서 bmi 정상인 비율이 약 70%의 수치를 가지며 정상이 아닌(저체중, 과체중, 비만) 비율의 합보다 항상 많은 것을 알 수 있다. 또한 정상 bmi가 아닌 경우 과체중 > 비만 > 저체중 순으로 비율을 차지하는 것을 알 수 있다. 전체의 경우 정상이 69.8%로 가장 많았고 저체중의 경우 4.95%로 가장 낮은 비율을 차지하고 있다. 남성의 경우 정상이 68.3%, 저체중이 4.25%로 전체 bmi 등급에서의 정상 비율과 저체중 비율보다 다소 낮은 모습을 보여준다. 반면 과체중, 비만에서는 전체 bmi 등급 비율에서보다 높은 비율을 보인다. 여성의 경우 정상이 71.5%로 가장 많은, 저체중이 5.72%로 가장 적은 비율을 보인다. 전체 bmi 등급 비율에서보다 정상, 저체중의 비율이 높게 보이며 과체중과 비만의 경우 다소 낮은 모습을 보인다.



[그림14] bmi 등급에 따른 Pie Chart

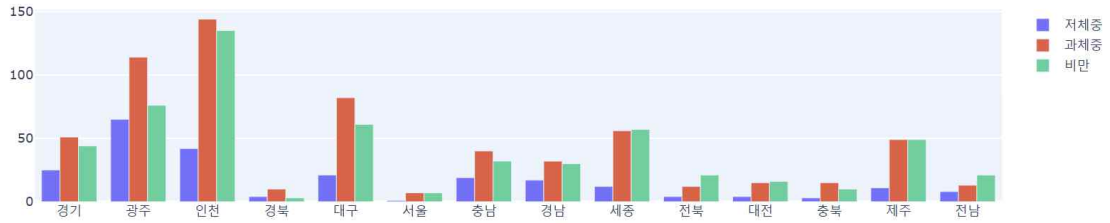
(2) Bar Chart

도시 규모에 따라 성별로 구분하여 bmi가 정상 범주가 아닌 등급을 [그림15]에서 막대그림으로 표현하였다. 대도시/중소도시의 남성의 경우 과체중에서 가장 많았고 저체중에서 가장 적었다. 과체중의 수가 저체중의 수보다 약 3.5배 정도 많았으며 비만의 수가 저체중의 수보다 약 3배 정도 많았다. 여성의 경우 또한 과체중에서 가장 많았고 저체중에서 가장 적었다. 과체중의 수가 저체중의 수보다 약 2배 많았으며 비만의 수가 저체중의 수보다 1.9배 많았다. 남성, 여성 포함하여 남성의 과체중이 가장 많았고 남성의 저체중이 가장 적은 것을 볼 수 있으며 남성의 저체중 대비 체중 과다의 비율이 여성의 저체중 대비 체중 과다의 비율보다 높다. 읍지역의 남성의 경우 비만이 가장 많았고 과체중이 가장 적었다. 과체중의 수가 저체중의 수보다 약 3.3배 많고 비만의 수가 저체중의 수보다 약 3.5배 많다. 여성의 경우 과체중이 가장 많았고 저체중이 가장 적었다. 과체중의 수가 저체중의 수보다 약 3배 많고 비만의 수가 저체중의 수보다 약 2배 많다. 대도시/중소도시의 과체중이 가장 많은 모습과는 다르게 읍지역의 남성의 경우 비만이 가장 많았다. 도서벽지/면지역의 남성의 경우 비만이 가장 많고 저체중이 가장 적었다. 과체중의 수가 저체중의 수보다 약 2배 많고 비만의 수가 저체중의 수보다 약 3배 많다. 여성의 경우 또한 비만이 가장 많고 저체중이 가장 적었다. 과체중의 수가 저체중의 수보다 약 2배 많고 비만의 수가 저체중의 수보다 약 3배 많다. 대도시/중소도시의 모든 성별에서 과체중이 가장 많은 모습과 비교했을 때 도서벽지/면지역은 모든 성별에서 비만이 가장 많다. 모든 지역에서 성별 상관없이 저체중의 수가 가장 적었으며 과다 체중의 수가 가장 많은 것을 알 수 있다.

[그림15] 성별, 도시 규모에 따른 비정상 bmi 등급 barplot

[그림16]은 시도에 따른 비정상 bmi 등급을 표현하였다. 시도는 경기, 광주, 인천, …, 제주, 전남으로 14개의 지역이 있는 것을 알 수 있다. 비정상 bmi 등급의 수가 가장 많은 지역은 인천이고 가장 적은 지역은 서울인 것을 알 수 있다. 서울의 인구가 가장 많음에도 서울의 비정상 bmi 등급의 수가 가장 적은 것은 결측값 처리 중 서울 데이터가 제거된 것으로 보인다.

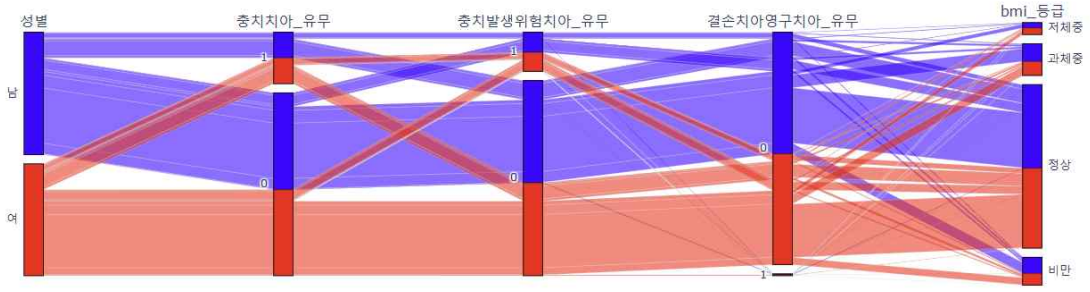
경기, 광주, 인천, 경북, 대구, 충남, 경남, 세종, 제주에서 과체중의 수가 가장 높고 비만, 저체중 순으로 저체중의 수가 가장 적은 것을 보인다. 특히 인천, 세종, 제주의 경우 저체중에 비해 과체중, 비만의 수가 눈에 띄게 많은 것을 알 수 있다. 반면 서울, 전북, 대전, 전남의 경우 비만, 과체중, 저체중의 순으로 비만의 수가 가장 많고 저체중의 수가 가장 적은 계단식의 모습을 보인다. 시도에 따른 학생 수는 차이가 있지만 시도에 따라 비정상 bmi의 비율에 차이가 있다고 하기 어렵다.



[그림16] 시도에 따른 비정상 bmi 등급 barplot

(3) Parallel sets Chart

[그림17]은 성별에 따라 이상 치아 유무에 따른 bmi 등급을 표현하였다. 이상 치아 중 충치 치아가 있는 수가 가장 많았으며 결손 치아, 영구 치아가 있는 수가 가장 적었다. 그림을 보면 성별에 따른 충치 치아 유무, 충치 발생 치아 유무, 결촌치아영구치아 유무, bmi_등급의 비율이 거의 반반인 것을 확인할 수 있다. 치아의 충치, 충치 발생 위험, 결촌 치아를 모두 가진 남성은 없고 여성의 경우 있는 것을 확인할 수 있다. 결손 치아, 영구 치아 유무의 경우 대부분이 없고 bmi 등급 분포 또한 고르게 분포되어 해당 열에 대한 분석은 의미가 없는 것을 알 수 있다. 남성이면서 충치 치아만 존재하고 다른 이상 치아가 없는 경우 대부분 bmi 등급이 정상에 분포하며 충치 치아가 있고 충치 발생 치아가 있는 경우 또한 bmi 등급이 대부분 정상인 값을 가진다. 여성이면서 충치 치아가 있고 다른 이상이 없는 경우 대부분 bmi 등급 정상에 분포하며 충치 치아가 있고 충치 발생 위험 치아가 있는 경우 또한 대부분 bmi 등급 정상에 분포하는 것을 보인다. 이를 통해 이상 치아 유무와 관계없이 bmi_등급에 고르게 분포하는 것을 확인할 수 있어 이상 치아는 bmi_등급에 큰 영향을 끼치지 않는다고 보인다.



[그림17] 성별에 따라 이상 치아의 유무에 따른 bmi 등급

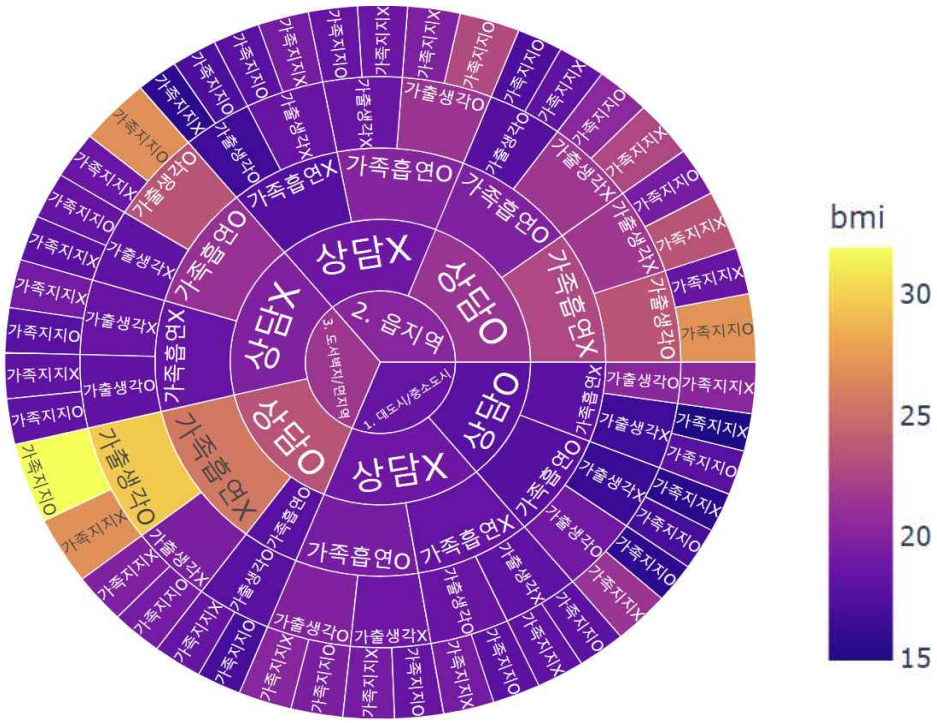
(4) Tree map Chart

[그림18]은 학교생활과 bmi를 시각화한 것으로 도시 규모별 학교생활에 따른 bmi를 Tree map으로 표현하였다. 도시 규모에 따라서는 도서벽지/면지역에서 bmi의 평균이 가장 높게 나타났고 대도시/중소도시에서 가장 낮게 나타났다. 대도시/중소도시에서 현금갈취가 있던 학생들의 bmi 평균은 20 이하였고 현금갈취가 없었던 학생들의 bmi 평균은 20 이상으로 보아 현금갈취는 덩치가 작은 학생들에게 주로 발생한 것으로 보인다. 현금갈취가 없는 학생 중 신체접촉의 경우 bmi의 평균이 높을수록 신체접촉 경험이 있었으며 평균이 낮을수록 신체접촉의 경험이 없는 것을 보인다. 현금갈취 경험이 없고 신체접촉이 있던 학생 중 괴롭힘이 없었던 학생에서 체벌 경험이 있는 학생의 bmi 평균이 체벌 경험이 없는 학생의 bmi 평균보다 높았으며 괴롭힘이 있던 학생에서는 체벌 경험이 없는 학생의 bmi 평균이 높음을 보인다. 도서벽지/면지역의 경우 현금갈취가 있는 학생의 bmi의 평균이 매우 높은 것으로 보이며 그 하위 속성들 또한 모두 높은 것으로 보인다. 현금갈취가 없는 학생 중 신체접촉 경험이 있을 때가 없을 때보다 평균 bmi가 높았으며 신체접촉이 없을 때 괴롭힘이 있는 학생이 괴롭힘이 없는 학생보다 bmi의 평균이 높은 것을 볼 수 있다. 도서벽지/면지역에서 현금갈취가 있는 학생, 현금갈취가 없으면서 신체접촉이 없고 괴롭힘당한 적이 있으며 체벌 경험이 있는 학생의 bmi 평균이 높았고 현금갈취가 없으며 신체접촉을 당했고 괴롭힘이 있던 학생의 bmi 평균이 높았다. 반면 현금갈취가 없으며 신체접촉 당한 적이 없고 괴롭힘당한 적 없으며 체벌 경험이 있는 학생의 경우 bmi의 평균이 낮은 것을 보인다. 읍지역의 경우 현금갈취를 당한 적 있는 학생이 없는 학생보다 bmi의 평균이 높았으며 현금갈취가 없고 신체접촉이 있는 학생의 bmi 평균이 낮았고 현금갈취가 없고 신체접촉이 없으며 괴롭힘이 없고 체벌 경험이 있는 학생의 bmi 평균이 다소 높은 것을 보인다. Tree map을 통해 현금갈취, 신체접촉, 괴롭힘, 체벌 경험에 따라 bmi의 평균에서 차이를 보이는 경우도 있으나 그렇지 않은 경우가 더 많아 해당 변수들은 bmi에 큰 영향을 주지 않는 것으로 판단된다.

[그림18] 도시 규모별 학교생활에 따른 bmi

(5) sunburst Chart

[그림19]는 도시 규모별 가정에 따른 bmi 평균을 sunburst Chart로 표현한 그림이다. 상담O, X는 각각 {상담O : 상담을 희망함, 상담X : 상담을 희망하지 않음}을 의미한다. 도서벽지/면지역의 경우 상담을 희망하는 학생이 희망하지 않는 학생보다 bmi 평균이 높은 것으로 보이며 상담을 희망하는 경우 가족 흡연을 할 때보다 가족 흡연을 하지 않을 때 bmi 평균이 높은 것을 보인다. 상담을 희망하며 가족이 흡연하지 않고 가출 생각이 있을 때 bmi 평균이 높게 나타났으며 특히 가출 생각이 있으면서 가족의 지지를 받을 때 지지를 받지 않을 때보다 bmi의 평균이 다소 높게 나타났다. 상담을 희망하지 않는 학생의 경우 가족 흡연을 할 때 하지 않을 때보다 bmi 평균이 다소 높게 나타났으며 가족 흡연을 하고 가출 생각이 있으며 가족의 지지를 받을 때 bmi 평균이 높게 나타남을 보인다. 상담 희망을 원하고 가족 흡연이 있을 때와 상담을 희망하지 않고 가족 흡연이 없을 땐 bmi의 평균이 대체로 20 이하인 것을 보인다. 읍지역의 경우 상담을 희망할 때 희망하지 않는 경우보다 bmi의 평균이 다소 높은 것을 보이며 가족이 흡연하지 않고 가출 생각이 있으며 가족의 지지를 받는 경우 bmi의 평균이 25 이상의 높은 수치를 보이며 가출 생각이 없고 가족 지지를 받지 않는 경우 bmi의 평균이 다소 높게 보인다. 상담 희망을 하며 가족 흡연이 있고 가출 생각이 있으며 가족 지지를 받을 때 bmi 평균이 가장 낮은 것을 볼 수 있다. 대도시/중소도시의 경우 상담 희망 여부와 상관없이 대부분에서 bmi의 평균이 낮게 나타났으며 그중 상담 희망을 하며 가족 흡연이 있고 가출 생각이 있으며 가족지지를 받지 않는 학생의 bmi가 23 정도로 가장 높은 평균을 보이고 있다. sunburst 분석을 통해 지역별로 상담 희망, 가족 흡연, 가출 생각, 가족지지에 따라 bmi 평균이 차이를 보이는 경우도 있으나 차이를 보이지 않는 경우도 있어 해당 열들이 bmi에 큰 영향을 주지 않는 것으로 판단된다.



[그림19] 도시 규모별 가정에 따른 bmi sunburst

4. Data Analysis

(1) Analysis

전처리를 수행한 학생 건강검사 데이터를 이용해 bmi_등급을 분류하는 모델을 생성한다. 분류분석을 위해 target 변수인 bmi_등급 변수를 LabelEncoder()를 통해 인코딩을 실시한다. {과체중 : 0, 비만 : 1, 저체중 : 2, 정상 3}으로 인코딩되었다.

전처리를 수행한 학생 건강검사 데이터를 7:3의 비율로 train, test 데이터 셋을 생성하고 train 데이터에서 훈련한 모델을 test 데이터에 적용한다. bmi_등급을 KNN, NA, SVM, DecisionTree, Randomforest, AdaBoost 분류기로 모델을 생성하였다.

KNN의 경우 K의 개수가 39일 때 분류 정확도가 가장 높았으며 분류 정확도는 0.7인 것을 보인다. [그림20]을 보면 K가 10 이하일 때 K가 증가할수록 분류 정확도가 급격하게 증가하는 것을 볼 수 있으나 분류 정확도의 최고치를 달성 후 K의 값이 증가하여도 분류 정확도는 증가하지 않는 것을 볼 수 있다. [그림23]의 KNN 분류 Confusion Matrix를 보면 실제로 과체중이지만 정상으로 분류한 값이 194, 실제로 비만이지만 과체중으로 분류한 값이 1과 정상으로 분류한 값이 177, 실제로 저체중이지만 정상으로 분류한 값이 55, 실제로 정상인데 정상으로 분류한 값이 1003으로 비만인데 과체중으로 분류한 값 1개를 제외하곤 모두 정상으로 분류한 것을 보인다. 정상의 수가 과체중, 저체중, 비만의 수보다 많아 모두 정상으로 분류해도 분류 정확도가 높게 나온 것으로 보인다. KNN 분류의 경우 분류 정확도는 다소 높지만 정상으로 편향된 분류를 수행하여 올바른 분류를 수행했다고 하기 어렵다.

NA의 분류 정확도는 0.079로 매우 낮으며 분류를 전혀 수행하지 못했다고 할 수 있다. [그림23]의 Confusion Matrix를 보면 실제로 과체중인 학생 194명 중 22명만 과체중으로 분류하고 22명은 비만, 150명은 저체중으로 오분류한 것을 볼 수 있다. 실제로 비만인 학생 178명 중 38명만 비만으로 분류하고 34명은 과체중, 105명은 저체중, 1명은 정상으로 오분류하였고, 실제로 저체중인 학생 55명중 51명을 저체중으로 분류하고, 2명을 과체중, 2명을 비만으로 오분류하여 저체중인 학생에 대한 분류는 잘 된 것으로 보인다. 실제로 정상인 학생은 1,003명 중 2명만 정상으로 분류되었고 52명이 과체중, 51명이 비만, 898명이 저체중으로 오분류되어 정상인 학생에 대한 분류가 제대로 수행되지 않은 것을 보인다. NA 분류의 경우 저체중으로 분류한 비율이 높아 저체중으로 편향된 분류를 수행한 것으로 보이고 분류 정확도 또한 매우 낮아 제대로 된 분류를 수행했다고 할 수 없다.

SVM의 분류 정확도는 0.724로 높은 분류 정확도를 보이고 있다. [그림23]의 Confusion Matrix를 보면 KNN과 비슷하게 대부분을 정상으로 분류한 것을 볼 수 있다. 실제로 과체중인 학생 194명 중 3명을 과체중으로 분류하고 13명을 비만, 178명을 정상으로 오분류하였고 실제로 비만인 학생 178명 중 38명을 비만으로 분류하였고 1명을 과체중, 139명을 정상으로 오분류하였다. 저체중인 학생은 55명 중 55명을 모두 정상으로 오분류하였고 정상인 학생 1003명 중 994명을 정상으로 분류하였고 2명을 과체중, 7명을 비만으로 오분류하였다. SVM의 경우 저체중에 대한 분류를 전혀 수행하지 못했지만 KNN과 다르게 거의 모든 값들을 정상으로 분류하지 않았고 분류 정확도 또한 증가하여 정상으로 분류하는 편향이 존재하는 것으로 보이지만 KNN에 비해 훌륭한 분류를 수행했다고 할 수 있다.

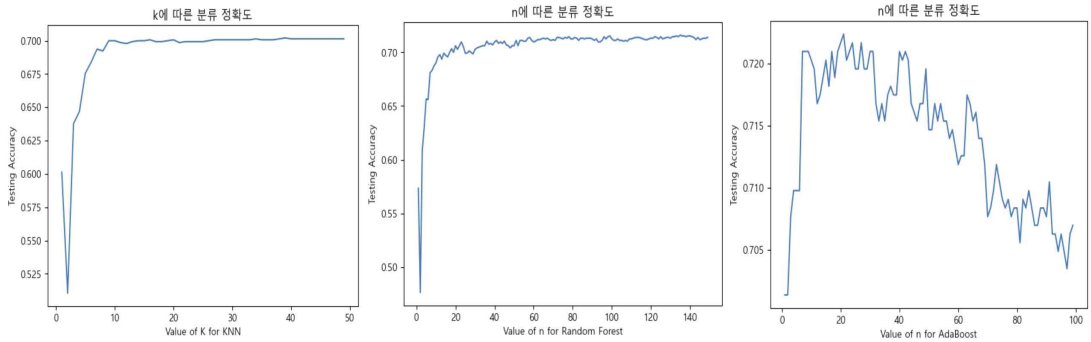
DecisionTree의 분류 정확도는 0.56으로 다소 낮은 분류 정확도를 보이고 있다. [그림23]의 Confusion Matrix를 보면 과체중인 학생 194명 중 31명을 과체중으로 분류하였고 43명을

비만, 9명을 저체중, 111명을 정상으로 오분류하였다. 비만인 학생 178명 중 52명을 비만으로 분류하였고 48명을 과체중, 2명을 저체중, 76명을 정상으로 오분류 하였다. 저체중인 학생 55명 중 6명을 저체중으로 분류하였고 과체중을 8명, 비만을 2명, 정상을 39명으로 오분류 하였다. 정상인 학생 1,003명 중 715명을 정상으로 분류하였고 과체중을 127명, 비만을 76명, 저체중을 85명으로 오분류 하였다.

[그림21]을 통해 Randomforest의 분류 정확도는 0.716으로 n이 135일 때 가장 높은 정확도를 보인다. [그림21]을 보면 n이 20 이하일 때 n이 증가할수록 급격하게 증가하다가 최고치에 달성한 후 n이 증가하더라도 정확도는 증가하지 않는 것을 보인다. [그림23]의 Confusion Matrix를 보면 과체중인 학생 중 6명을 정분류 하였고 22명을 비만, 166명을 정상으로 오분류 하였다. 비만인 학생 중 37명을 정분류 하였고 14명을 과체중, 127명을 정상으로 오분류 하였다. 저체중의 경우 1명을 과체중으로, 54명을 정상으로 오분류하여 저체중에 대한 분류를 전혀 수행하지 못하였다. 정상인 학생 중 978명을 정분류 하였고 10명을 과체중, 15명을 비만으로 오분류하였다. Confusion Matrix를 통해 랜덤포레스트 분류는 SVM과 마찬가지로 저체중에 대한 분류를 전혀 수행하지 못한 것을 볼 수 있다. 분류 정확도는 SVM보다 다소 떨어져 아쉬운 결과를 보인다.

마지막으로 AdaBoost의 분류 정확도는 0.722이고 n이 21일 때 가장 높은 정확도를 보인다. [그림22]를 보면 n이 30 이하일 때 n이 증가할수록 분류 정확도가 증가하였는데 n이 31 이상인 경우 n이 증가할수록 오히려 분류 정확도가 급격하게 감소하는 것을 볼 수 있다. [그림23]의 Confusion Matrix를 보면 과체중인 학생 중 33명을 정분류 하였고 31명을 비만, 2명을 저체중, 128명을 정상으로 오분류 하였다. 비만인 학생 중 61명을 정분류 하였고 26명을 과체중, 1명을 저체중, 90명을 정상으로 오분류 하였다. 저체중인 학생 중 1명을 정분류 하였고 1명을 과체중, 2명을 비만, 51명을 정상으로 오분류 하였다. 정상인 학생 중 916명을 정분류 하였고 37명을 과체중, 29명을 비만, 21명을 저체중으로 오분류 하였다. AdaBoost에서도 저체중에 대한 분류가 다소 잘 이루어지지 않은 것을 보인다.

[그림20] K에 따른 KNN 분류 정확도 [그림21] n에 따른 Randomforest 분류 정확도 [그림22] n에 따른 AdaBoost 분류 정확도




```
[[   0    0    0  194]
 [   1    0    0  177]
 [   0    0    0   55]
 [   0    0    0 1003]]
n_estimators : 39
KNN 분류 정확도 Max : 0.7020979020979021
```

```
[[ 22  22 150   0]
 [ 34  38 105   1]
 [  2   2  51   0]
 [ 52  51 898   2]]
NA 분류 정확도: 0.07902097902097902
```

```
[3 3 3 ... 3 3 3]
[[  3  13   0 178]
 [  1  38   0 139]
 [  0   0   0  55]
 [  2   7   0 994]]
SVC 분류 정확도 : 0.7237762237762237
```

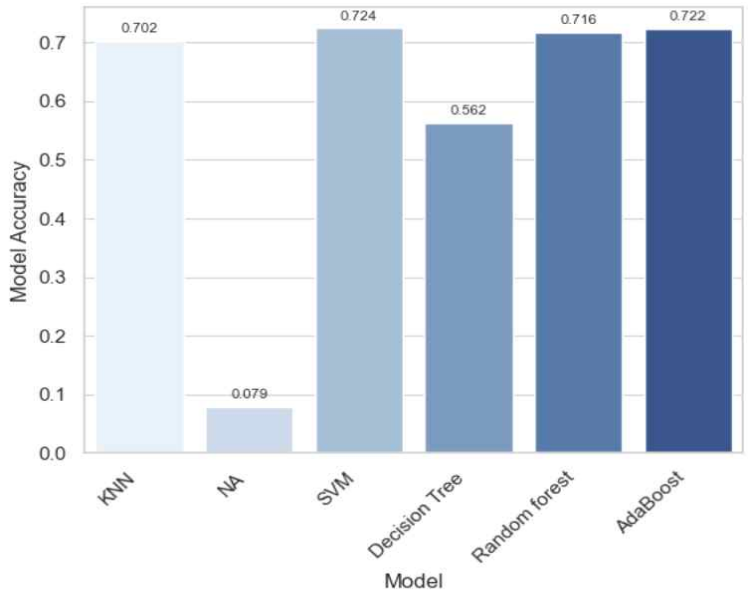
```
Confusion Matrix:  [[ 31  43   9 111]
 [ 48  52   2  76]
 [  8   2   6  39]
 [127  76  85 715]]
DecisionTree 분류 정확도: 0.5622377622377622
```

```
Confusion Matrix:  [[  6  22   0 166]
 [ 14  37   0 127]
 [  1   0   0  54]
 [ 10  15   0 978]]
Randomforest 분류 정확도 : 0.7160839160839161
n_estimators : 135
```

```
Confusion Matrix:  [[ 33  31   2 128]
 [ 26  61   1  90]
 [  1   2   1  51]
 [ 37  29  21 916]]
ada 분류 정확도 : 0.7223776223776224
n_estimators : 21
```

[그림23] 모든 열들을 이용한 KNN, NA, SVM, DecisionTree, Randomforest, AdaBoost 분류

6가지의 모델을 이용해 분류를 실시하였다. [그림23]을 이용하여 [그림24]에서 소수점 3자리까지 정확도를 표현하였고 이를 통해 분류 정확도의 경우 SVM 모델에서 0.724로 가장 높았고 NA 모델에서 0.079로 가장 낮음을 알 수 있다. 분류 정확도 측면에서는 SVM 모델이 가장 우수하다고 할 수 있지만 [그림23]의 Confusion Matrix를 통해 SVM의 경우 저체중에 대한 분류를 전혀 수행하지 못하고 정상에 대한 분류만을 수행하여 정확도가 높다고 판단되고 AdaBoost의 경우 저체중에 대한 분류도 수행하여 정확도는 SVM에 비해 다소 낮지만 AdaBoost의 분류가 더 우수하다고 판단된다.

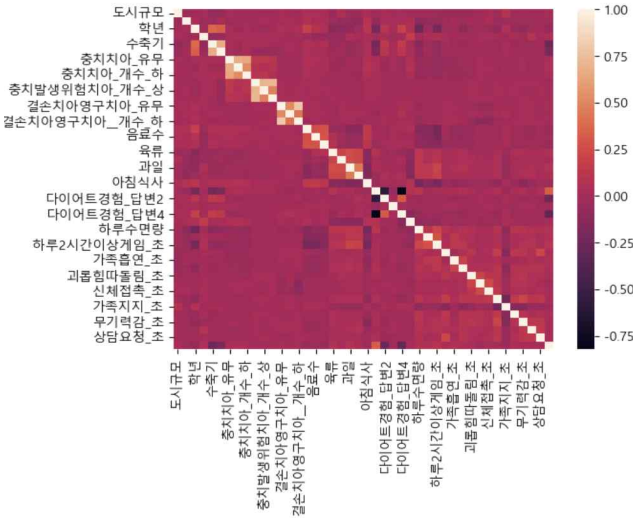


[그림24] 모든 열을 이용한 모델의 분류 정확도 비교

(2) Improve Accuracy

학생 건강검사 데이터의 [그림25] heatmap을 그려보았다. 해당 데이터의 열들이 많아 bmi_등급_encoded와의 상관관계가 높은 열들을 확인하기가 어렵다. 위의 그림을 보면 2~4개의 열을 제외하고는 대부분 bmi_등급_encoded와 상관관계가 없는 것으로 보인다. bmi_등급_encoded와 상관관계가 0.15 이상이거나 -0.15 이하인 값들만을 추출하였고 '수축기', '이완기', '다이어트경험_답변1', '다이어트경험_답변2', '다이어트경험_답변3' 열들을 이용해 train, test 데이터 셋을 7:3 비율로 나눠 분류 모델을 생성하였다.

(2) - 1. 상관관계 높은 열 추출




```
수축기                -0.257355
이완기                -0.190172
다이어트경험_답변1       0.336779
다이어트경험_답변2      -0.254063
다이어트경험_답변4      -0.297066
bmi_등급_encoded    1.000000
Name: bmi_등급_encoded, dtype: float64
```

[그림25] 열들 간의 상관관계 및 상관관계 높은 열 추출

KNN의 분류 정확도는 0.7223으로 k가 43일 때 가장 높은 분류 정확도를 가졌으며 모든 데이터를 이용해 분류를 수행했을 때보다 약 0.02 정도 분류 정확도가 증가한 것을 볼 수 있다. [그림26]을 보면 K가 증가할수록 분류 정확도가 약간의 증가세를 보이는 것을 알 수 있고 모든 데이터에 대한 KNN 분류를 수행했을 때 거의 모든 데이터를 정상으로 분류하여 과체중, 저체중, 비만에 대한 분류가 잘 이루어지지 않았는데 상관관계가 높은 열들을 이용할 경우 저체중을 제외한 과체중, 비만에 대한 분류가 잘 수행된 것을 볼 수 있다.

NA의 분류 정확도의 경우 0.67로 모든 데이터를 이용해 분류를 수행했을 때보다 약 0.6 분류 정확도가 증가하며 분류가 이전보다 훌륭하게 수행된 것을 볼 수 있다. 모든 데이터를 이용했을 땐 비만, 과체중, 저체중에 대한 분류만 수행되고 정상에 대한 분류가 수행이 안 됐었는데 상관관계가 높은 열들을 이용했을 때 정상에 대한 분류가 잘 수행되어 NA 분류의 경우 상관관계가 낮은 불필요한 열들이 많을 경우 분류를 제대로 수행하지 못하는 것을 알 수 있다.

SVM의 분류 정확도의 경우 0.7로 모든 데이터를 이용해 분류를 수행했을 때보다 약 0.02 감소하여 분류 모델 성능이 더 안 좋아진 것을 볼 수 있다. 모든 데이터를 이용했을 때 저체중에 대한 분류는 수행하지 못했지만 비만, 과체중, 정상에 대한 분류는 수행한 것으로 확인되었는데 상관관계가 낮은 열들을 제거했을 땐 비만, 저체중, 과체중에 대한 분류를 전혀 수

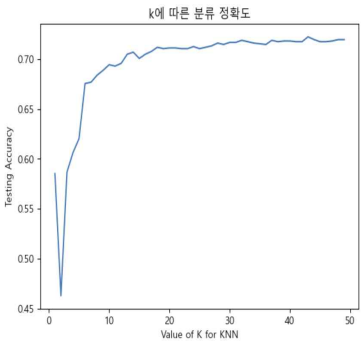
행하지 못하고 모든 데이터를 정상으로 분류한 것을 볼 수 있다. SVM의 경우 target의 개수와 열의 개수에 영향을 많이 받는다고 예상할 수 있다.

Decision Tree의 분류 정확도는 0.64로 모든 데이터를 이용해 분류를 수행했을 때보다 약 0.07 증가하여 모델 성능이 좋아진 것을 알 수 있다. 하지만 [그림29]를 보면 모든 데이터를 사용했을 땐 저체중인 학생을 6명 정분류 하였는데 열 선택 후 저체중을 모두 오분류한 것을 볼 수 있다. 그 대신 정상인 학생을 약 100명 정도 정분류하여 분류 정확도가 상승한 것으로 보인다.

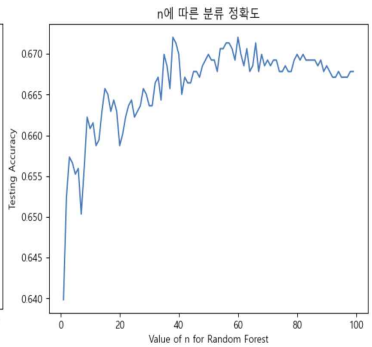
Random forest의 분류 정확도는 n이 38일 때 가장 높았으며 0.67로 모든 데이터를 사용했을 때보다 약 0.05 감소하며 모델 성능이 안 좋아진 것을 볼 수 있다. [그림27]을 보면 n이 증가할수록 분류 정확도가 증가와 감소를 반복하다가 n이 70 이상일 때부터 감소 추세를 보이며 모든 데이터를 썼을 때 최고치에서 증가 혹은 유지하는 모습과 다름을 보인다. Random forest에서도 저체중을 정분류한 값은 없고 과체중, 비만으로 오분류한 사례가 많아진 것을 볼 수 있다.

Ada Boost의 [그림28] 분류 정확도를 보면 n이 5일 때 0.72로 가장 큰 정확도를 가지며 모든 데이터를 이용해 분류를 수행했을 때보다 0.003 정도 감소한 것을 볼 수 있다. 정확도가 크게 감소한 것이 아니기 때문에 성능이 감소했다고 보긴 어렵지만 이전의 분류에서는 저체중을 1개 정분류 했었는데 열 선택 후 저체중에 대한 분류가 전혀 수행되지 않은 것을 알 수 있다. [그림28]을 보아 n이 증가할수록 정확도는 감소세를 보인다.

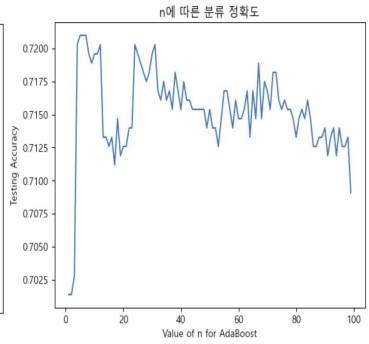
[그림26] K에 따른 KNN 분류 정확도



[그림27] n에 따른 Randomforest 분류 정확도



[그림28] n에 따른 AdaBoost 분류 정확도




```
[[  9  24   0 161]
 [  8  44   0 126]
 [  0   1   0  54]
 [  3  24   0 976]]
n_estimators : 43
KNN 분류 정확도 Max : 0.7223776223776224
```

```
[[ 36  55   3 100]
 [ 27  90   0  61]
 [  1   1   0  53]
 [ 65  85  21 832]]
NA 분류 정확도: 0.66993006993007
```

```
[[   0    0    0  194]
 [   0    0    0  178]
 [   0    0    0   55]
 [   0    0    0 1003]]
SVC 분류 정확도 : 0.7013986013986014
```

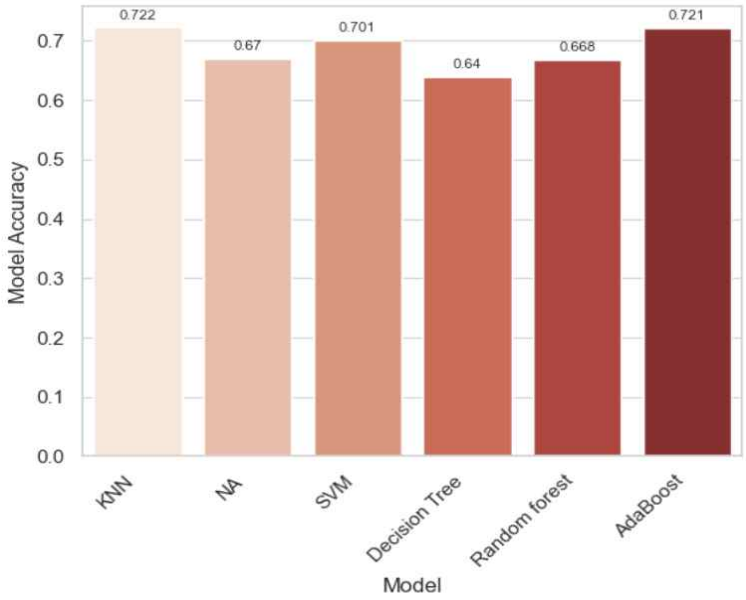
```
Confusion Matrix:  [[ 29  36   4 125]
 [ 47  43   1  87]
 [  4   2   0  49]
 [ 80  51  29 843]]
Decision Tree 분류 정확도: 0.6398601398601399
```

```
Confusion Matrix:  [[ 28  29   1 136]
 [ 30  40   1 107]
 [  2   1   0  52]
 [ 50  47  19 887]]
n_estimators : 38
랜덤포레스트 분류 정확도: 0.6678321678321678
```

```
Confusion Matrix:  [[ 10  19   1 164]
 [ 17  27   0 134]
 [  0   1   0  54]
 [  9  17   0 977]]
ada 분류 정확도 : 0.720979020979021
n_estimators : 5
```

[그림29] 상관관계가 높은 열들 KNN, NA, SVM, DecisionTree, Randomforest, AdaBoost 분류

[그림29]를 통해 [그림30]으로 상관관계가 높은 열들로 분류를 수행한 모델들의 분류 정확도를 소수점 세 자리까지 표현하여 막대그래프로 나타냈다. 6가지 모델 중 KNN에서 분류 정확도가 가장 높았고 Decision Tree에서 분류 정확도가 가장 낮았다. Ada Boost는 모든 열을 썼을 때와 마찬가지로 높은 분류 정확도를 보이고 NA 분류의 분류 정확도는 확연히 증가한 것을 그림으로 볼 수 있다. Decision Tree 또한 증가하였지만 NA에 비해 증가세가 적고 다른 분류 모델에 비해 정확도가 떨어짐을 보인다.



[그림30] 상관관계 높은 열을 이용한 모델의 분류 정확도 비교

(2) - 2. 모델 Stacking

분류 정확도 향상을 위해 모델 Stacking 방법을 실시하였다. 단일 모델에서 우수한 성능을 보였던 SVM, KNN, Ada Boost 모델을 선택하였다. [그림31]을 보면 4개의 Stacking 모델을 수행했음을 알 수 있는데 먼저 SVM과 KNN 분류 모델을 Stacking 하여 Decision Tree 분류를 파악한다. 분류 정확도는 0.58로 기존의 단일 SVM, KNN 모델보다 정확도가 감소한 것을 볼 수 있으나 정상에 대한 분류가 잘 이루어지지 않고 비만, 과체중, 저체중에 대한 분류가 단일 모델에 비해 잘 이루어졌음을 확인할 수 있다.

SVM과 KNN 분류 모델을 Stacking 한 후 위와 다르게 Logistic 분류를 수행하였다. 분류 정확도는 0.723으로 단일 모델 분류 정확도가 KNN보다 상승하였고 SVM보다 다소 감소하였다. Decision Tree와 비교했을 때 분류 정확도가 약 0.14 증가하며 Logistic을 사용했을 때 모델의 성능이 향상하는 것을 보인다. 다만 정상에 대한 분류는 잘 수행됐지만 저체중, 과체중에 대한 분류는 전혀 수행되지 않은 것을 볼 수 있다.

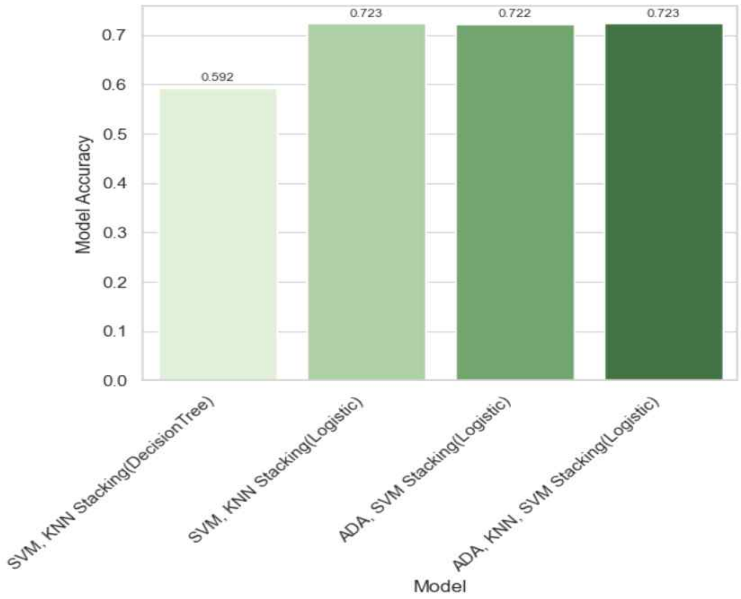
AdaBoost와 SVM 분류 모델을 Stacking 하여 Logistic 분류를 수행하였다. 분류 정확도는 0.72로 단일 모델 SVM보다 정확도가 감소한 것을 보이며 SVM, KNN Stacking에 비해 분류 정확도가 0.01 감소한 것을 보인다. Ada, SVM Stacking 모델 또한 과체중과 저체중에 대한

분류를 전혀 수행하지 못하며 비만과 정상에 대한 분류만을 수행하는 것을 볼 수 있다.
AdaBoost, KNN, SVM 분류 모델을 Stacking 하여 Logistic 분류를 수행하였다. 분류 정확도는 0.723인 단일 SVM 모델보다 분류 정확도가 다소 감소하였고 SVM, KNN Stacking 모델과의 Confusion Matrix를 비교했을 때 동일한 분류와 분류 정확도를 가지는 것을 보인다. 이 모델 또한 과체중과 저체중에 대한 분류를 전혀 수행하지 못하는 것을 볼 수 있다.

```
Confusion Matrix:  [[ 37  33   8 116]
 [ 43  56   6  73]
 [  3   3   6  43]
 [113  67  89 734]]
SVM, KNN 스태킹(DecisionTree) 분류 정확도: 0.5825174825174825

Confusion Matrix:  [[  0  17   0 177]
 [  0  40   0 138]
 [  0   0   0  55]
 [  0   9   0 994]]
SVM, KNN 스태킹(Logistic) 분류 정확도: 0.7230769230769231

Confusion Matrix:  [[  0  16   0 178]
 [  0  39   0 139]
 [  0   0   0  55]
 [  0   9   0 994]]
ADA, SVM 스태킹(Logistic) 분류 정확도: 0.7223776223776224

Confusion Matrix:  [[  0  17   0 177]
 [  0  40   0 138]
 [  0   0   0  55]
 [  0   9   0 994]]
ADA, KNN, SVM 스태킹(Logistic) 분류 정확도: 0.7230769230769231
```

[그림31] 모델 Stacking

[그림31]을 통해 [그림32]로 모델에 따라 분류 정확도를 소수점 3자리까지 표현하며 비교하였다. SVM, KNN Stacking(Logistic)과 ADA, KNN, SVM Stacking(Logistic)의 모델에서 분류 정확도가 0.723으로 가장 높았으며 SVM, KNN Stacking(Decision Tree)의 모델에서 분류 정확도가 0.592로 가장 낮았다. 해당 데이터의 분류 Logistic 분류가 Decision Tree 분류보다 우수한 성능을 보였고 모델 Stacking을 했을 때 단일 모델 SVM보다는 정확도가 다소 감소한 것을 보여준다.



[그림32] Stacking 후 모델의 분류 정화도 비교

(3) - 3. Target ratio

성능 향상을 위해 상관관계가 높은 열 추출, 스태킹 방법을 시도하였지만 분류 정확도에 차이가 없음을 볼 수 있고 대부분의 모델에서 정상에서만 분류가 잘되고 비만, 과체중, 저체중에 대한 분류는 잘 수행하지 못하는 것을 볼 수 있다. 이 문제가 target 비율의 불균형으로 인해 발생했다고 판단하여 RandomSampling, Smote 방법으로 이를 해결하여 분류를 수행한다.

```
Confusion Matrix:  [[ 18  35   2 139]
 [ 31  44   1 102]
 [  2   0   0  53]
 [ 32  29   3 939]]
Randomforest 분류 정확도 : 0.7034965034965035
n_estimators : 113
```

```
[[ 61  39  45  49]
 [ 72  48  34  24]
 [ 12   3  21  19]
 [240 114 336 313]]
n_estimators : 1
KNN 분류 정확도 Max : 0.6013986013986014
```

```
Confusion Matrix:  [[ 67  59  31  37]
 [ 60  88  12  18]
 [  3   5  30  17]
 [193 118 383 309]]
ada 분류 정확도 : 0.37902097902097903
n_estimators : 2
```

[그림] random oversampling 후 분류 정확도

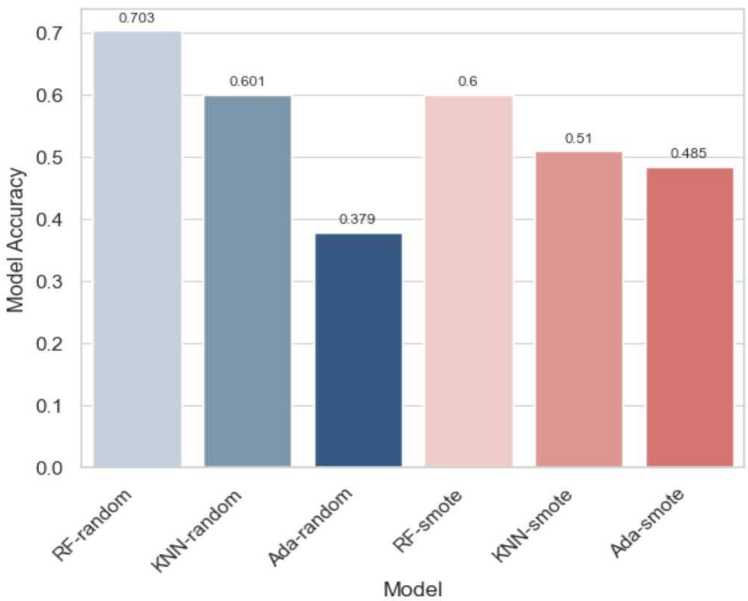
```
Confusion Matrix:  [[ 45  42   4 103]
 [ 53  48   5  72]
 [  4   2   4  45]
 [113  47  82 761]]
Randomforest 분류 정확도 : 0.6
n_estimators : 14
```

```
[[ 64  34  43  53]
 [ 75  45  33  25]
 [ 11   3  23  18]
 [217  97 351 338]]
n_estimators : 1
KNN 분류 정확도 Max : 0.5097902097902098
```

```
Confusion Matrix:  [[ 41  43  27  83]
 [ 32  73  11  62]
 [  6   3  22  24]
 [128  65 286 524]]
ada 분류 정확도 : 0.4853146853146853
n_estimators : 37
```

[그림] SMOTE 후 분류 정확도

RandomForest, KNN, AdaBoost에 각각 적용하여 정확도를 비교하였다. target의 불균형 비율을 해소하여 훈련하였더니 오히려 분류 정확도가 낮아졌음을 알 수 있다. RandomSampling, Smote 방법을 사용하면 target의 불균형을 해소할 수 있지만 과적합 문제 등이 발생할 수 있다. 해당 데이터는 target의 불균형으로 인해 정확도가 낮은 것이 아닌 것으로 보인다. 따라서 RandomSampling, Smote 방법은 사용하지 않는다.



[그림] target 불균형 비율 해소 후 정확도 비교

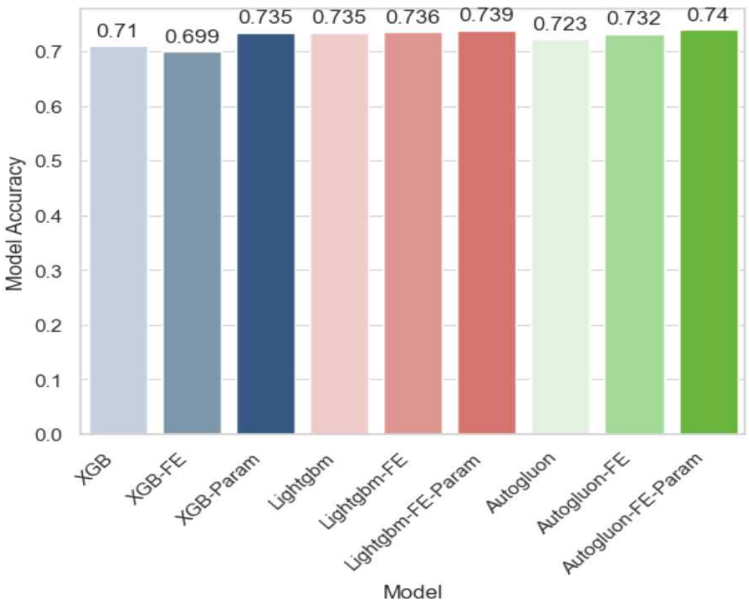
(4) - 4. Adding Models & feature engineering & Optimizing Hyper Parameters

기존의 6가지 모델 외의 모델을 적용한다. Kaggle 및 다양한 대회에서 우수한 모델로 손꼽히고 있는 XGB, Lightgbm과 AWS에서 개발한 예측 모델인 autogluon을 사용하여 예측을 수행한다. 그 후 feature engineering을 통해 변수를 추가, 삭제 과정을 진행한다. 마지막으로 optuna를 이용해 하이퍼 파라미터를 최적화한다.

[표] 파생변수 추가

수축기*이완기	수축기 × 이완기
수축기2	수축기 × 수축기
이완기2	이완기 × 이완기
다이어트경험	다이어트경험_답변1 + … + 다이어트경험_답변4
괴롭힘	괴롭힘따돌림_초 + 현금갈취_초 + 신체접촉_초
나쁜식단	라면 + 음료수 + 패스트푸드
좋은식단	채소(김치제외) + 과일
학교생활	체벌경험_초+수업태도교정_초
개인심리	무기력감_초+상담요청_초

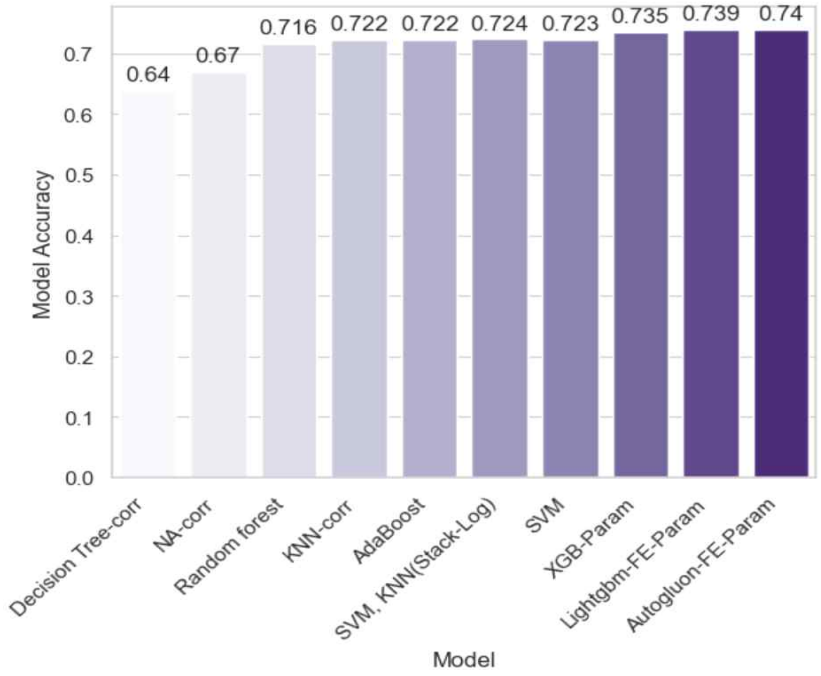
XGB, Lightgbm, Autogluon에 대해 분류를 수행하였다. feature engineering을 진행하기 전 분류 정확도는 Lightgbm에서 0.735로 가장 높았고 XGB에서 0.71로 가장 낮았다. feature engineering을 한 뒤 각 모델에 따라 분류를 다시 수행하였을 때 Lightgbm, Autogluon에서는 정확도가 증가하였으나 XGB에서는 감소한 것을 볼 수 있다. 모든 모델에서 하이퍼 파라미터를 최적화했을 때 정확도가 상승하였고 Hyper Parameter가 정확도에 큰 영향을 미침을 알 수 있다. 전체 모델 중 feature engineering을 수행하고 하이퍼 파라미터를 최적화한 Autogluon의 정확도가 가장 높았고 feature engineering만을 수행한 XGB가 가장 낮았다.



[그림] 추가 모델들의 feature engineering, Hyper Parameter 최적화 전과 후 정확도 비교

마지막으로 각 모델 중 분류 정확도가 가장 높은 모델을 가져와 비교하였다. 모델 중 feature engineering을 수행하고 하이퍼 파라미터를 최적화한 Autogluon 모델의 분류 정확도가 가장 높았고 상관관계가 높은 열을 추출한 Decision Tree 모델이 가장 낮았다. 상위 3개 모델은 나중에 추가한 XGB, Lightgbm, Autogluon으로, 기존에 수업 시간에 배우던 KNN부터 Adaboost 모델보다 XGB, Lightgbm, Autogluon 모델의 성능이 더 훌륭하다는 것을 알 수 있다.

해당 데이터에 대해 다양한 모델에 적용하고 target 비율을 맞춰 훈련하고 feature engineering을 진행하였지만 모델의 정확도는 크게 상승하지 않은 것을 알 수 있다. 이는 해당 데이터에 많은 열들이 있지만 학생 bmi와 관련이 있는 열이 거의 없어서 발생한 모습으로 보인다. 그럼에도 열들의 숨은 관계를 찾아내고 분류 정확도를 증가시키는 일이 데이터 분석에 있어서 중요함을 알 수 있었다.



[그림] 각 모델 중 분류 정확도가 가장 높은 모델들 비교