변영철 교수님

인공지능 머신러닝 과제

수면 지표 변동에 따른 보행 패턴 상관 관계 탐색

컴퓨터공학과
2021108260
안예원

목차

개요

대학생의 수면의 양과 질이 대인일탈 행동에 미치는 영향: 일기연구를 이용한 부정정서의 매개효과 검증

The Effect of Sleep Quantity and Quality on College Students' Interpersonal Deviant Behavior: A Daily Diary Examination of Mediating Role of Negative Affect

전수면박탈이 정상인의 미세운동수행 능력에 미치는 영향

전수면박탈이 정상인의 미세운동수행 능력에 미치는 영향

동향·연구보고서

내 학술정보에 저장

잠을 제대로 못 잤다는 '생각'만으로도 업무 수행 능력이 떨어질 수 있다

24시간 수면 박탈이 최대 운동 후 Interleukine-2의 변화와 운동수행에 미치는 영향

Effects of 24 hours` sleep deprivation on the changes of interleukine-2 and exercise performance after maximum exercise

개요

수면 지표 변동에 따른
보행 패턴 상관 관계 탐색

관련 연구

전수면박탈이 정상인의 미세운동수행 능력에 미치는 영향

전수면박탈이 정상인의 미세운동수행 능력에 미치는 영향

<연구 내용>

1. 오른손잡이 피 실험자 대상으로 38시간(첫날 오전 6시 ~ 다음날 오후 8시) 수면 박탈
2. 최대한의 속도로 펜을 두드리는 tapping 검사에서 오른손 기능 저하 발견
3. 수면박탈에 의한 근육의 피로가 쉽게 나타남을 확인

본 과제에서는 복합운동인 보행과 수면의 연관에 대해 탐구

내용 요약

수면 시간과 보행 패턴은 유의미한 상관 관계를 가진다

지도 학습 과정

1. 데이터셋 로드
2. 결측값 분석
3. 데이터 분석
4. 모델 훈련
5. 예측 결과

지도 학습 과정 (1) 데이터셋 로드

1. 데이터셋 로드를 위한 라이브러리 불러오기
2. 데이터셋 로드 및 이를 확인할 수 있는 함수 만들기
3. MyData 라는 클래스로 묶기

```
import pandas as pd
import seaborn as sns
import numpy as np


class MyData:
    data= 0
    #엑셀 파일 읽어오기
    def load_data(self, src):
        self.data = pd.read_excel(src)
    #파일 헤더 보여주기
    def show_head(self, backgroundColor, fontColor) :
        self.data.head().style.set_properties(**{'background-color':backgroundColor,'color':fontColor,'border-color':'#8b8c8c'})
    def show_info(self):
        self.data.info()
    def replace_NaN_with(self,value):
        self.data.fillna(value)
```

지도 학습 과정 (1) 데이터셋 로드

+ Code + Markdown

```
gildong = MyData()
gildong.load_data('/kaggle/input/prior-nights-sleep-and-gait/20210318_gait_data_1.xlsx')
gildong.show_head('royalblue','white')
```

	Subjectnumber	Age	Gender1male2female	Heightcm	WeightinKG	BMI	Hours_slept_prior_night	lessthan7_yes1_no2	morethan9_yes1_no2	7to9_yes1_no2	7to9_yes1_no2
0	24	20	2	167.640000	47.727273	16.945592	7.750000	2	2	1	Tru
1	324	20	1	175.260000	73.636364	23.920605	4.750000	1	2	2	Fals
2	361	20	2	165.100000	83.181818	30.449467	2.400000	1	2	2	Fals
3	801	20	2	170.180000	71.363636	24.586990	7.167000	2	2	1	Tru
4	911	21	1	175.260000	68.000000	22.089645	8.080000	2	2	1	Tru

지도 학습 과정 (1) 데이터셋 로드

```
gildong.show_info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 126 entries, 0 to 125
Data columns (total 84 columns):
 #   Column                                              Non-Null Count  Dtype
---  ------                                              --------------  -----
 0   Subjectnumber                                       126 non-null    int64
 1   Age                                                 126 non-null    int64
 2   Gender1male2female                                  126 non-null    int64
 3   Heightcm                                            125 non-null    float64
 4   WeightinKG                                          125 non-null    float64
 5   BMI                                                 125 non-null    float64
 6   Hours_slept_prior_night                             126 non-null    float64
 7   lessthan7_yes1_no2                                  126 non-null    int64
 8   morethan9_yes1_no2                                  126 non-null    int64
 9   7to9_yes1_no2                                       126 non-null    int64
 10  7to9_yes1_no2.1                                     126 non-null    bool
 11  PlusMinus1_typical_Yes1_No2                         0 non-null      float64
 12  PlusMinus1_typical_Yes1_No2.1                       0 non-null      float64
 13  Anticipatory_Postural_Adjustment_Durations          110 non-null    float64
 14  Anticipatory_Postural_Adjustment_First_Step_Durations 110 non-null  float64
 15  Anticipatory_Postural_Adjustment_First_Step_ROM     110 non-null    float64
 16  Anticipatory_Postural_Adjustment_Forward_Peak       110 non-null    float64
 17  Anticipatory_Postural_Adjustment_Lateral_Peak       110 non-null    float64
 18  Back_Right_Frontal_Bending_Max                      124 non-null    float64
 19  Back_Left_Frontal_Bending_Max                       124 non-null    float64
 20  Back_Frontal_Bending_ROM                            124 non-null    float64
 21  Back_Sagittal_Max_Angle                             124 non-null    float64
 22  Back_Sagittal_Min_Angle                             124 non-null    float64
 23  Back_Sagittal_ROM                                   124 non-null    float64
 24  Back_Transverse_Right_Max_Angle                     124 non-null    float64
 25  Back_Transverse_Left_Max_Angle                      124 non-null    float64
 26  Back_Transverse_ROM                                 124 non-null    float64
 27  Neck_Right_Frontal_Bending_Max                      113 non-null    float64
 28  Neck_Left_Frontal_Bending_Max                       113 non-null    float64
 29  Neck_Frontal_Bending_ROM                            113 non-null    float64
 30  Neck_Sagittal_Max_Angle                             113 non-null    float64
 31  Neck_Sagittal_Min_Angle                             113 non-null    float64
 32  Neck_Sagittal_ROM                                   113 non-null    float64
 33  Neck_Transverse_Right_Max_Angle                     113 non-null    float64
 34  Neck_Transverse_Left_Max_Angle                      113 non-null    float64
 35  Unnamed: 35                                         113 non-null    float64
 36  Neck_Transverse_ROM                                 113 non-null    float64
 37  Cadence                                             125 non-null    float64
```

- **Subjectnumber** : 피실험자 번호
- **Age** : 나이
- **Gender1male2female** : 성별 (1: 남성, 2: 여성)
- **Heightcm** : 키 (센티미터)
- **WeightinKG** : 체중 (킬로그램)
- **BMI** : 체질량지수
- **Hours_slept_prior_night** : 전날 밤 잔 시간
- **lessthan7_yes1_no2** : 7시간 미만 수면 여부 (1: 네, 2: 아니오)
- **morethan9_yes1_no2** : 9시간 이상 수면 여부 (1: 네, 2: 아니오)
- **7to9_yes1_no2** : 7~9시간 수면 여부 (1: 네, 2: 아니오)
- **7to9_yes1_no2.1** : 7~9시간 수면 여부 (boolean)
- **그 외의 열** : 데이터 집합의 다양한 신체 측정값 및 걷기/운동 동작에 대한 정보들

지도 학습 과정 (2) 결측값 분석

결측치 -> 분석 왜곡, 모델 예측 정확성에 영향

1) 데이터셋에 결측치가 있는 지 확인
2) 결측치 처리

지도 학습 과정 (2) 결측값 분석

```
import missingno as msno

class MissingValueShow:
    fix = 0
    ax = 0
    axs = 0
    def set_sub_plot(self,x,y) :
        self.fig, self.ax = plt.subplots(2,2,figsize=(12,7))
        self.axs = np.ravel(self.ax)
    def show_matrix(self, data, fontsize, value) :
        msno.matrix(data, fontsize=fontsize, color=(0.25,0,0.5), ax = value);
    def show_bar(self, data, fontsize, value):
        msno.bar(data, fontsize=fontsize, color=(0.25,0,0.5), ax= value);
    def show_heatmap(self, data, fontsize, value):
        msno.heatmap(data,fontsize=fontsize,ax = value)
    def show_dendrogram(self, data, fontsize, value) :
        msno.dendrogram(data,fontsize=fontsize,ax=value, orientation='top')
```

1. 결측치 시각화 라이브러리(missingno) 불러오기

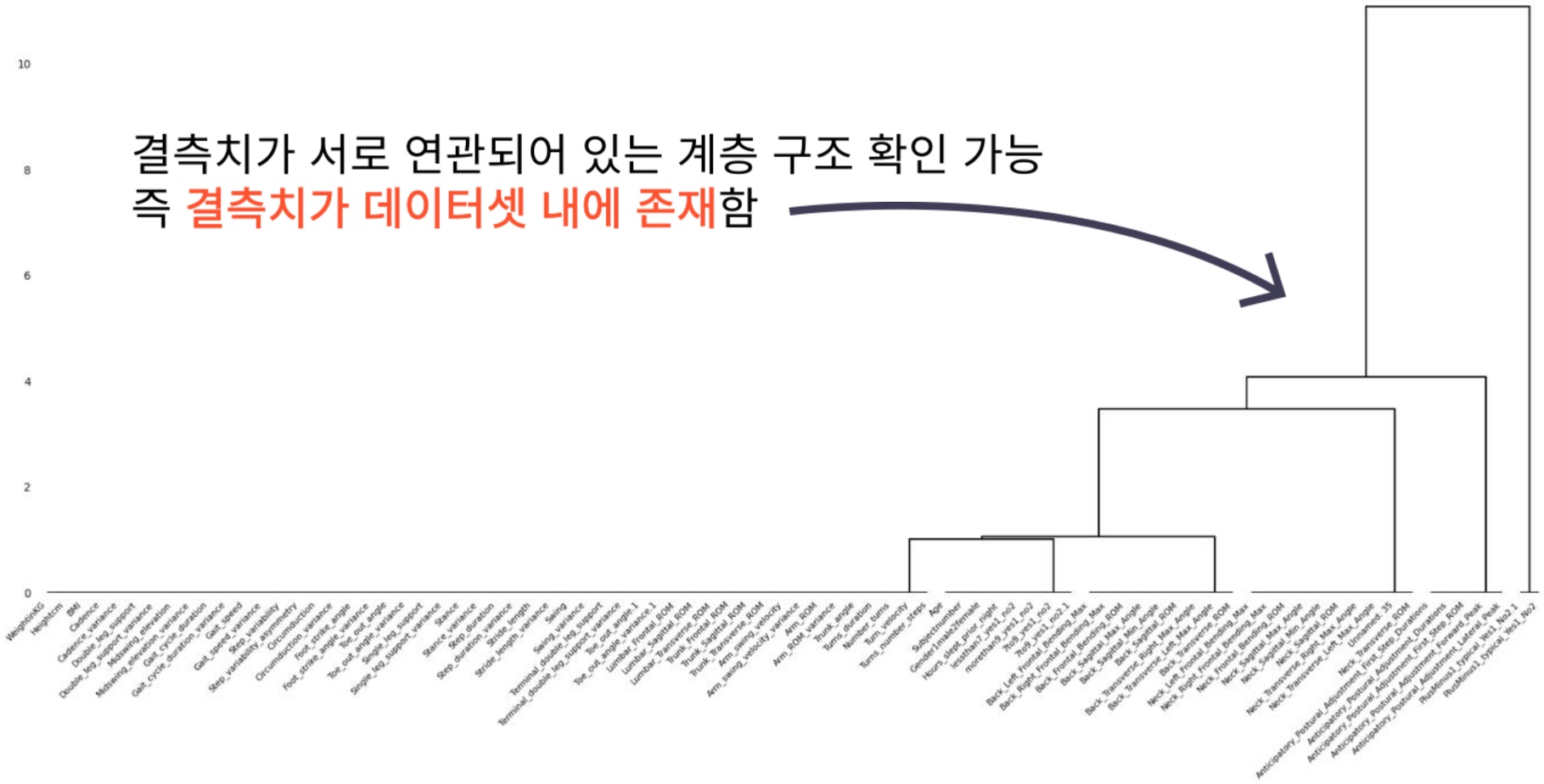
2. 결측치를 확인할 수 있는 함수 만들기

3. MissingValueShow라는 클래스로 묶기

지도 학습 과정 (2) 결측값 분석

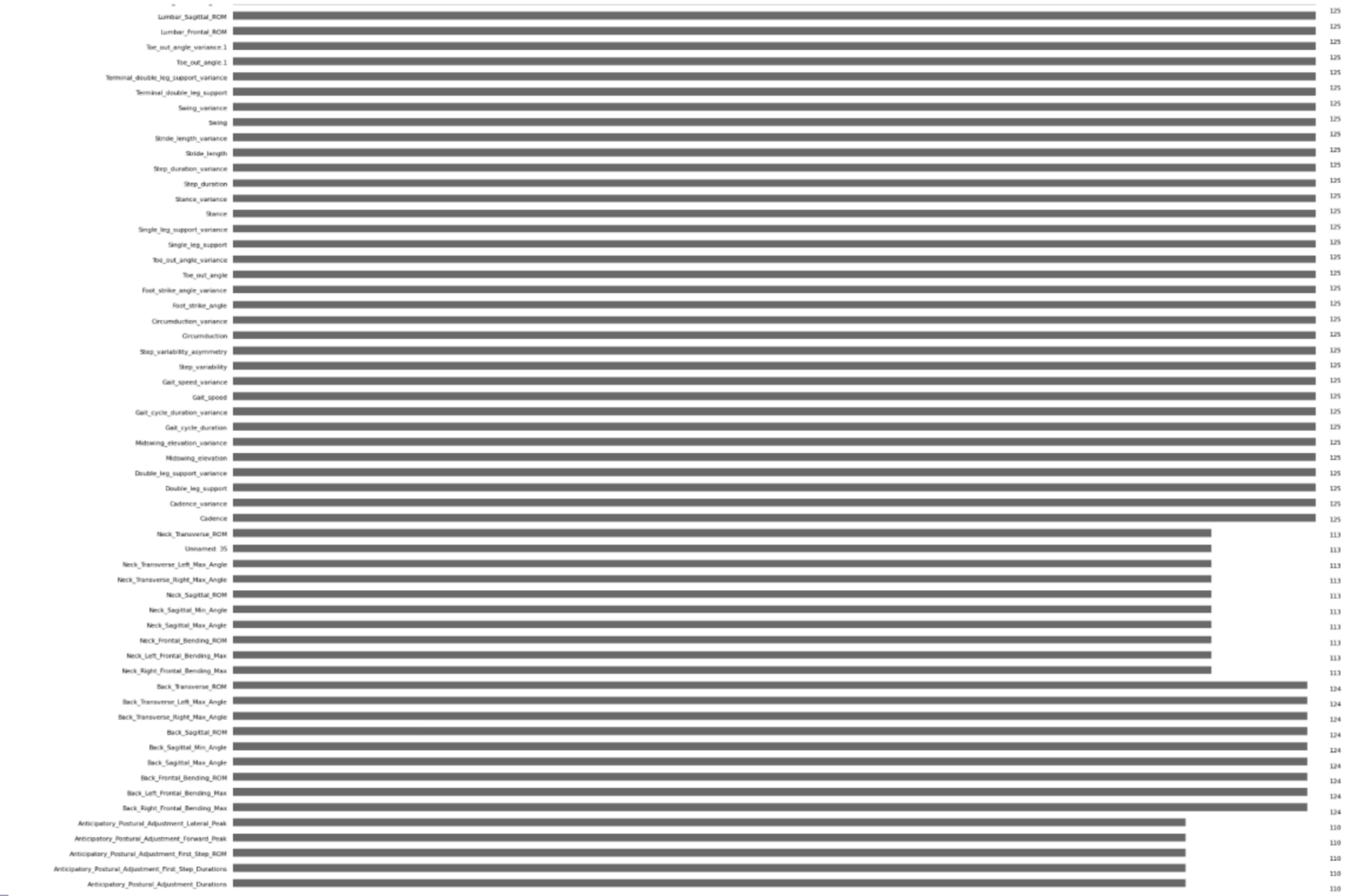
```
cheolsu.show_dendrogram(gildong.data, 8)
```

```
<Axes: >
```



지도 학습 과정 (2) 결측값 분석

```
cheolsu.show_bar(gildong.data, 8)
```



막대 그래프를 통해서도
결측치 확인

지도 학습 과정 (2) 결측값 분석

<결측치가 존재하는 컬럼>

- 목의 최소 움직임
- 목의 최대 움직임
- 등이 왼쪽으로 최대 굽히는 정도
- 등이 오른쪽으로 최대 굽히는 정도
- 예상 자세 조정 처음 단계 시간
- 예상 자세 조정 첫 번째 단계 ROM
- 등의 최소 굴곡
- 등의 최대 굴곡
- BMI

```
# 결측값 처리
gildong.replace_NaN_with(0)
```

지도 학습 과정 (3) 데이터 분석

```python
import matplotlib.pyplot as plt
import plotly.express as px

class Analysis:
    custom_style = {
    'font': {'family': 'Arial', 'size': 16, 'color': 'black'},
    'title': {'font': {'size': 24, 'color': 'green'}},  # 폰트랑 색깔 미리 커스텀해서 전역 변수로 사용
    }
    def show_age_distribution(self,data):
        fig1 = px.histogram(data, x='Age', title='Age Distribution')
        fig1.update_layout(template='none', **self.custom_style)  #커스텀된 전역 변수 사용
        fig1.show()
    def show_height_distribution(self,data) :
        fig7 = px.histogram(data, x='Heightcm', title='Height Distribution')
        fig7.update_layout(template='plotly_white', **self.custom_style)
        fig7.show()
    def show_gender_distribution(self, data) :
        gender_counts = data['Gender1male2female'].value_counts()
        fig2 = px.pie(names=gender_counts.index, values=gender_counts.values, title='Gender Distribution')
        fig2.update_traces(textinfo='percent+label')
        fig2.update_layout(template='none', **self.custom_style)
        fig2.show()
    def show_sleptHoursPriorNight_distribution(self, data):
        fig4 = px.histogram(data, x='Hours_slept_prior_night', title='Hours Slept Prior Night Distribution')
        fig4.update_layout(template='none', **self.custom_style)
        fig4.show()
    def show_sleptHoursPrioirNightByGender_distribution(self, data):
        fig9 = px.box(data, x='Gender1male2female', y='Hours_slept_prior_night', points="all", title='Hours Slept Prior Night by Gender')
        fig9.update_layout(template='plotly_white', **self.custom_style)
        fig9.show()
    def show_anticiparyPosturalAdjustment_duration(self, data):
        fig12 = px.line(data, x=data.index, y='Anticipatory_Postural_Adjustment_Durations', title='Anticipatory Postural Adjustment Durations')
        fig12.update_layout(template='plotly_white', **self.custom_style)
        fig12.show()
    def show_gait_speed(self, data):
        fig13 = px.line(data, x=data.index, y='Gait_speed', title='Gait Speed')
        fig13.update_layout(template='plotly_white', **self.custom_style)
        fig13.show()
    def show_genderByBMI_distribution(self, data) :
        fig10 = px.box(data, x='Gender1male2female', y='BMI', points="all", title='BMI by Gender')
        fig10.update_layout(template='plotly_white', **self.custom_style)
        fig10.show()
```

1. 데이터 분석을 위한 라이브러리 불러오기

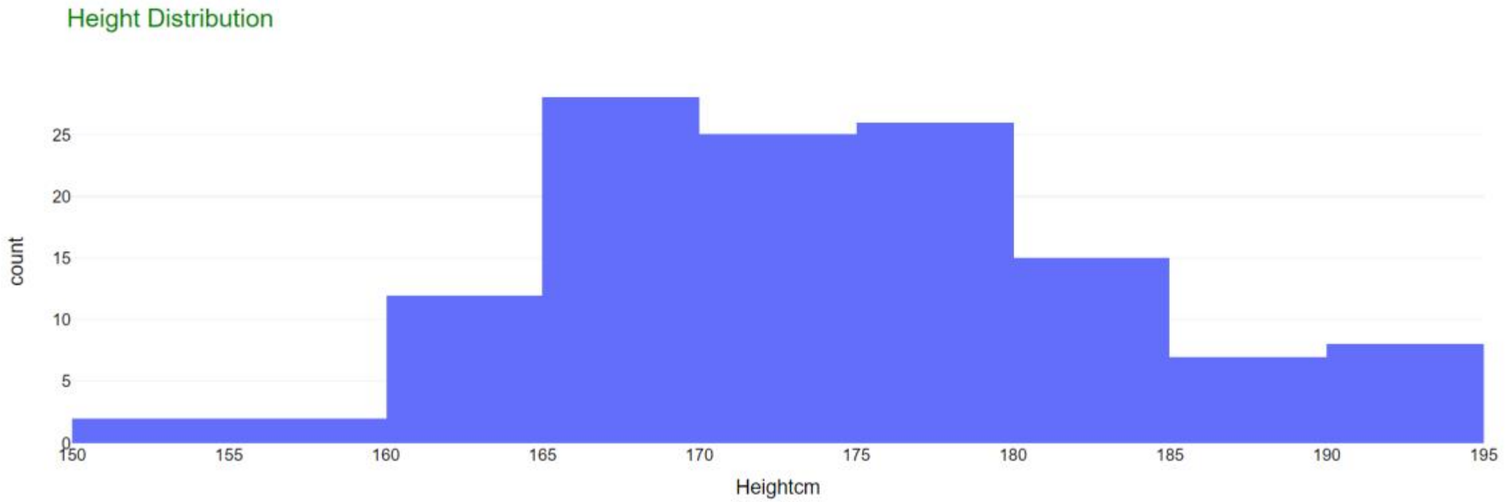
2. 분석하고 싶은 컬럼에 대한 함수 만들기

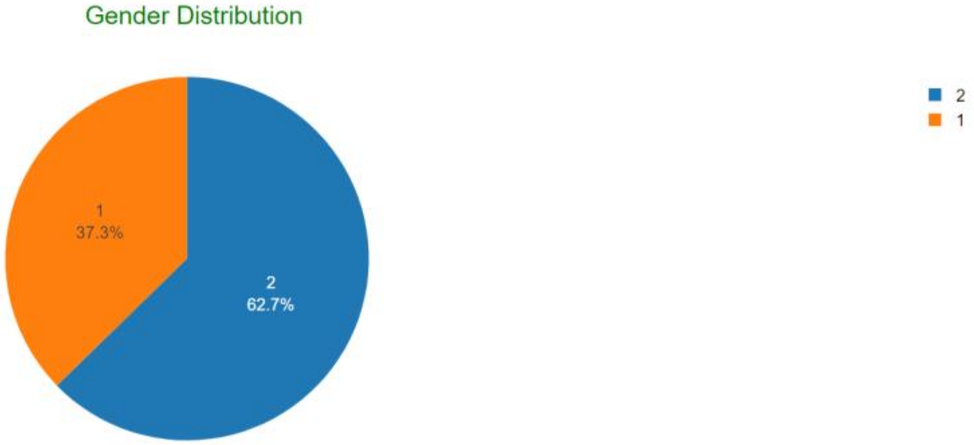
3. Analysis라는 클래스로 묶기

지도 학습 과정 (3) 데이터 분석

```
cheolsu = Analysis()

#데이터셋 내 키 분포 확인
cheolsu.show_height_distribution(gildong.data)
#데이터셋 내 성별 분포 확인
cheolsu.show_gender_distribution(gildong.data)
```





- 키

: 150~195까지 분포

- 성별

: 여성 데이터가 많음

지도 학습 과정 (3) 데이터 분석

```
#성별 기준으로 전 날 수면시간이 어떻게 되는지 시각화
cheolsu.show_sleptHoursPrioirNightByGender_distribution(gildong.data)
#성별 기준으로 BMI 분포가 어떻게 되는지 시각화
cheolsu.show_genderByBMI_distribution(gildong.data)
```

- 전 날 수면 시간

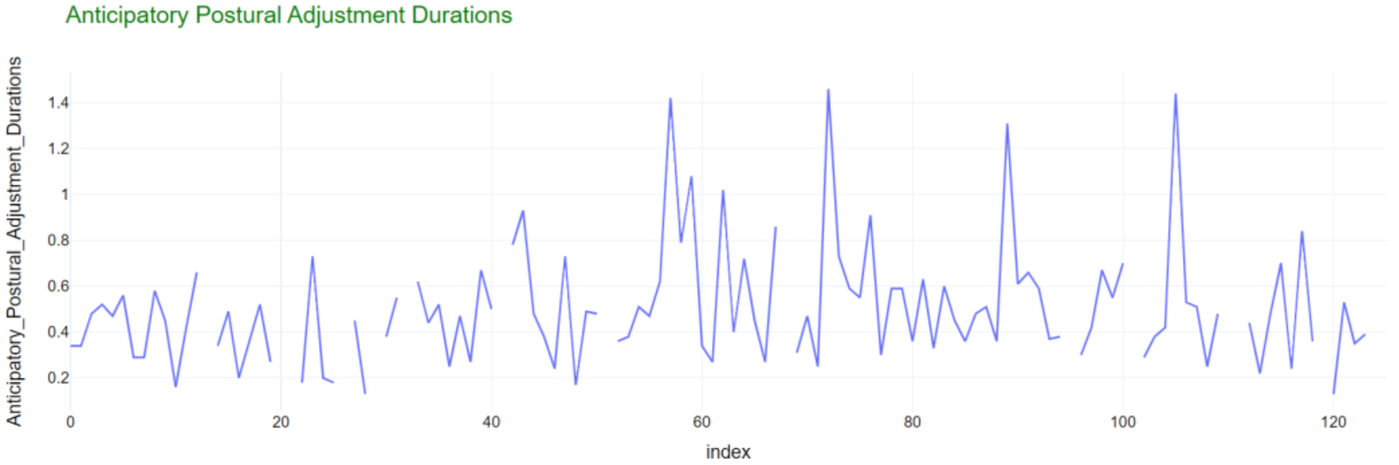
: 남성은 4~10시간 사이로 고르게 분포
여성은 주로 6~9시간 사이 내로 분포

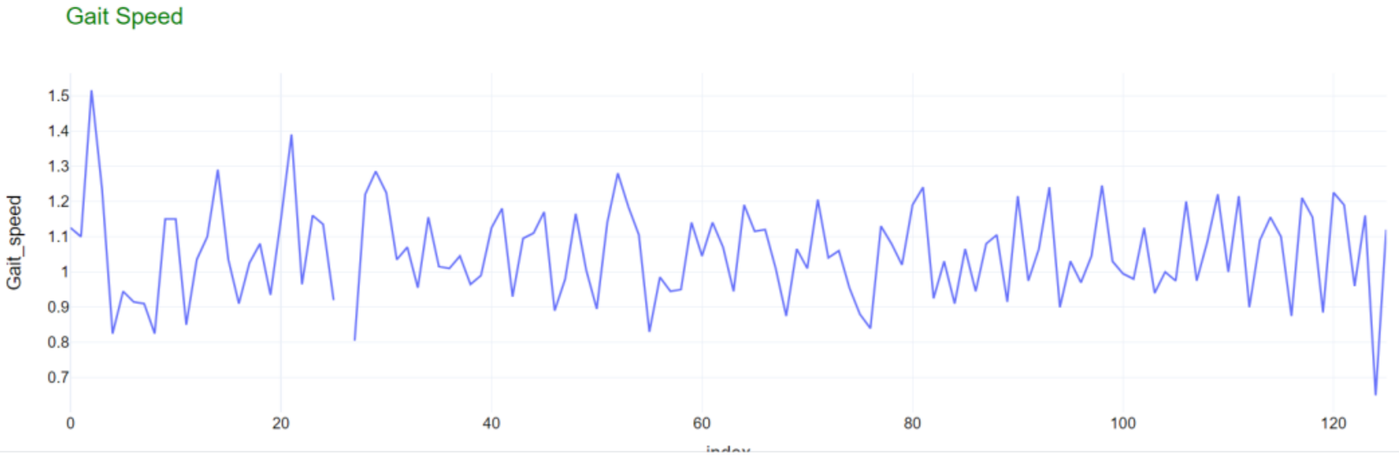
- BMI

: 양 측 대부분 표준에 위치

지도 학습 과정 (3) 데이터 분석

```
#예상 자세 조정 중에 취한 첫 걸음의 지속 시간을 나타냄 -> 즉 자극에 얼마나 빨리 반응하는 지
cheolsu.show_anticiparyPosturalAdjustment_duration(gildong.data)
#보행 속도 시각화
cheolsu.show_gait_speed(gildong.data)
```





- 자극 반응 시간

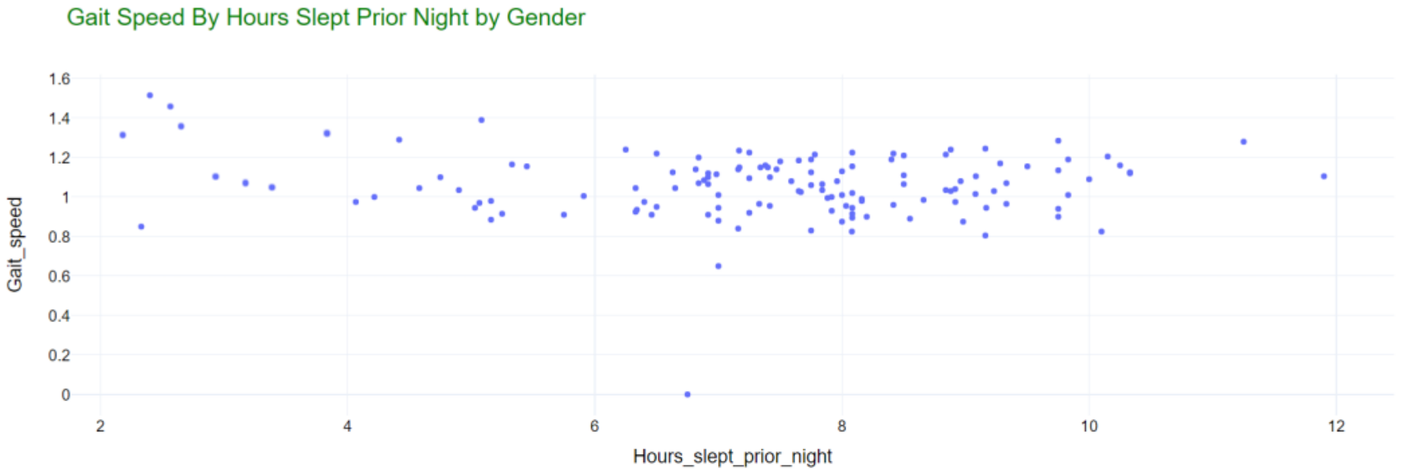
: 0.4 ~ 1.4초 대까지 다양
평균 0.5초 대로 자극에 반응

- 보행 속도

: 평균 1m/s

지도 학습 과정 (3) 데이터 분석

```
#전 날 수면 시간에 따른 보행 속도 시각화
cheolsu.show_gaitSpeedBySleptHours(gildong.data)
#전 날 수면 시간에 따른 자극 반응 속도 시각화
cheolsu.show_anticiparyPosturalAdjustmentBySleptHours(gildong.data)
```



- 전 날 수면 시간에 따른 보행 속도

: 7~9 시간을 숙면한 경우 평균 보행 속도 (1m/s) 가깝게 분포.
수면 시간이 떨어질 수록 평균을 상위하는 값 존재. 이상치(약 1.5) 발견

- 전 날 수면 시간에 따른 자극 반응 속도

: 수면 시간이 떨어질 수록 자극에 반응하는 속도가 느린 데이터 분포 확인

지도 학습 과정 (4) 모델 훈련

```python
class MyAI:
    X = 0
    y = 0
    X_train = 0
    X_test = 0
    y_train = 0
    y_test = 0
    def set_features(self, x) :
        self.X = x
    def set_target(self, y):
        self.y = y
    def set_test(self, size, random) :
        self.X_train, self.X_test, self.y_train, self.y_test = train_test_split(X, y, test_size = size, random_state = random)
    def run_LR(self) : #논리 회귀 알고리즘
        model = LinearRegression()
        model.fit(self.X_train, self.y_train)
        y_pred =model.predict(self.X_test)
        mse = mean_squared_error(self.y_test, y_pred)
        print(f'LinearRegression MSE: {mse:.2f}')

    def run_RF(self) : #랜덤 포레스트 알고리즘
        model = RandomForestRegressor()
        model.fit(self.X_train, self.y_train)
        y_pred =model.predict(self.X_test)
        mse = mean_squared_error(self.y_test, y_pred)
        print(f'RandomForestRegressor MSE: {mse:.2f}')

    def run_SVR(self) : #서포트 벡터 머신
        model = SVR()
        model.fit(self.X_train, self.y_train)
        y_pred =model.predict(self.X_test)
        mse = mean_squared_error(self.y_test, y_pred)
        print(f'SVR MSE: {mse:.2f}')


    def run_GB(self):#그래디언트 부스팅
        model = GradientBoostingRegressor()
        model.fit(self.X_train, self.y_train)
        y_pred =model.predict(self.X_test)
        mse = mean_squared_error(self.y_test, y_pred)
        print(f'GradientBoostingRegressor MSE: {mse:.2f}')


    def run_KNN(self): #K-근접 알고리즘
        model = KNeighborsRegressor()
        model.fit(self.X_train, self.y_train)
        y_pred =model.predict(self.X_test)
        mse = mean_squared_error(self.y_test, y_pred)
        print(f'KNeighborsRegressor MSE: {mse:.2f}')
```

모델의 훈련 결과는 MSE 지표로 표시한다
(MSE : 모델의 예측값과 실제 값 사이의 평균적인 오차의 제곱값)
-> MSE 값이 작을수록 모델의 예측값과 실제 값 사이의 차이가 적다는 것을 의미.
즉 작은 MSE는 일반적으로 더 나은 모델

1. 다양한 분류 알고리즘 불러오기
   (선형 회귀, KNN, 써포트 벡터 머신 등)
2. 훈련 함수 만들기
   - 학습용, 훈련용으로 데이터 나누기 함수
   - 모델 훈련 함수
3. MyAI라는 클래스로 묶기

지도 학습 과정 (4) 모델 훈련

목표 : 나이, 키, 몸무게, BMI, 전 날 밤의 수면 시간에 따른 보행 속도를 예측한다

```
youngja = MyAI()
youngja.set_features(gildong.data[['Age', 'Heightcm', 'WeightinKG', 'BMI', 'Hours_slept_prior_night']])
youngja.set_target(gildong.data['Gait_speed'])
youngja.set_test(0.2, 42)
youngja.run_LR()
youngja.run_RF()
youngja.run_SVR()
youngja.run_GB()
youngja.run_KNN()
```

입력 변수

입력 변수에 대해
예측할 값

```
LinearRegression MSE: 0.14
RandomForestRegressor MSE: 0.06
SVR MSE: 0.07
GradientBoostingRegressor MSE: 0.08
KNeighborsRegressor MSE: 0.06
```

전체 데이터의 20% 훈련용으로 할당,
데이터를 무작위로 섞는 시드값 42 할당

지도 학습 과정 (4) 모델 훈련

결과 : 랜덤 포레스트와 K-근접이 0.06의 낮은 MSE로 목표에 부합하는 알고리즘임을 보임

```
youngja = MyAI()
youngja.set_features(gildong.data[['Age', 'Heightcm', 'WeightinKG', 'BMI', 'Hours_slept_prior_night']])
youngja.set_target(gildong.data['Gait_speed'])
youngja.set_test(0.2, 42)
youngja.run_LR()
youngja.run_RF()
youngja.run_SVR()
youngja.run_GB()
youngja.run_KNN()
```

```
LinearRegression MSE: 0.14
RandomForestRegressor MSE: 0.06
SVR MSE: 0.07
GradientBoostingRegressor MSE: 0.08
KNeighborsRegressor MSE: 0.06
```

지도 학습 과정 (4) 모델 훈련

1) 랜덤 포레스트 테스트 해보기

```
best_model = youngja.run_RF()
sample_input = [[25, 170, 70, 24, 8]]  #테스트
predicted_gait_speed = best_model.predict(sample_input)
print(f'Predicted Gait_speed: {predicted_gait_speed[0]:.2f}')
```

```
Predicted Gait_speed: 1.10
```

- 나이, 키, 몸무게, 체질량 지수 동일하게 유지
- 수면 시간 8과 3으로 테스트

=> 보행 시간이 1.10에서 1.34로 늘어남을 확인

```
best_model = youngja.run_RF()
sample_input = [[25, 170, 70, 24, 3]]  #테스트
predicted_gait_speed = best_model.predict(sample_input)
print(f'Predicted Gait_speed: {predicted_gait_speed[0]:.2f}')
```

```
Predicted Gait_speed: 1.34
```

지도 학습 과정 (4) 모델 훈련

2) KNN 테스트 해보기

```
best_model = youngja.run_KNN()
sample_input = [[25, 170, 70, 24, 8]]  # 테스트
predicted_gait_speed = best_model.predict(sample_input)
print(f'Predicted Gait_speed: {predicted_gait_speed[0]:.2f}')
```

```
Predicted Gait_speed: 1.10
```

```
best_model = youngja.run_KNN()
sample_input = [[25, 170, 70, 24, 3]]  # 테스트
predicted_gait_speed = best_model.predict(sample_input)
print(f'Predicted Gait_speed: {predicted_gait_speed[0]:.2f}')
```

```
Predicted Gait_speed: 1.13
```

- 나이, 키, 몸무게, 체질량 지수 동일하게 유지
- 수면 시간 8과 3으로 테스트

=> 보행 시간이 1.10에서 1.13으로 늘어남을 확인

결론

전 날 수면 시간이 적을수록 보행 속도가
감소하는 패턴이 관찰된다

맺음말

탐구 과제에 여러 함수를 사용하여
시각화된 결과 및
유의미한 결과를 관찰하는 과정을 통해
인공지능 분야에 관심을 갖게 되었습니다