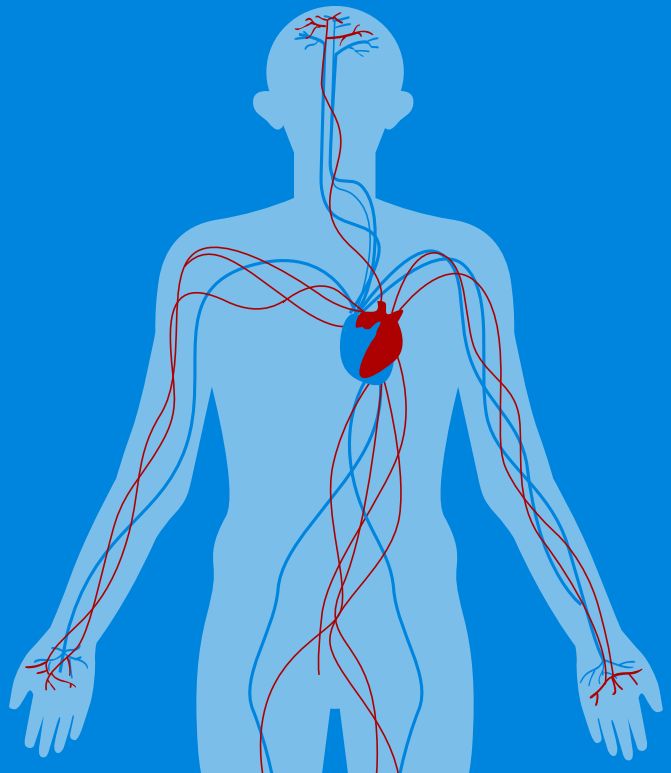
심혈관계 질환 예측 프로젝트

2022-07-08 ~ 07-11

이현민

Contents



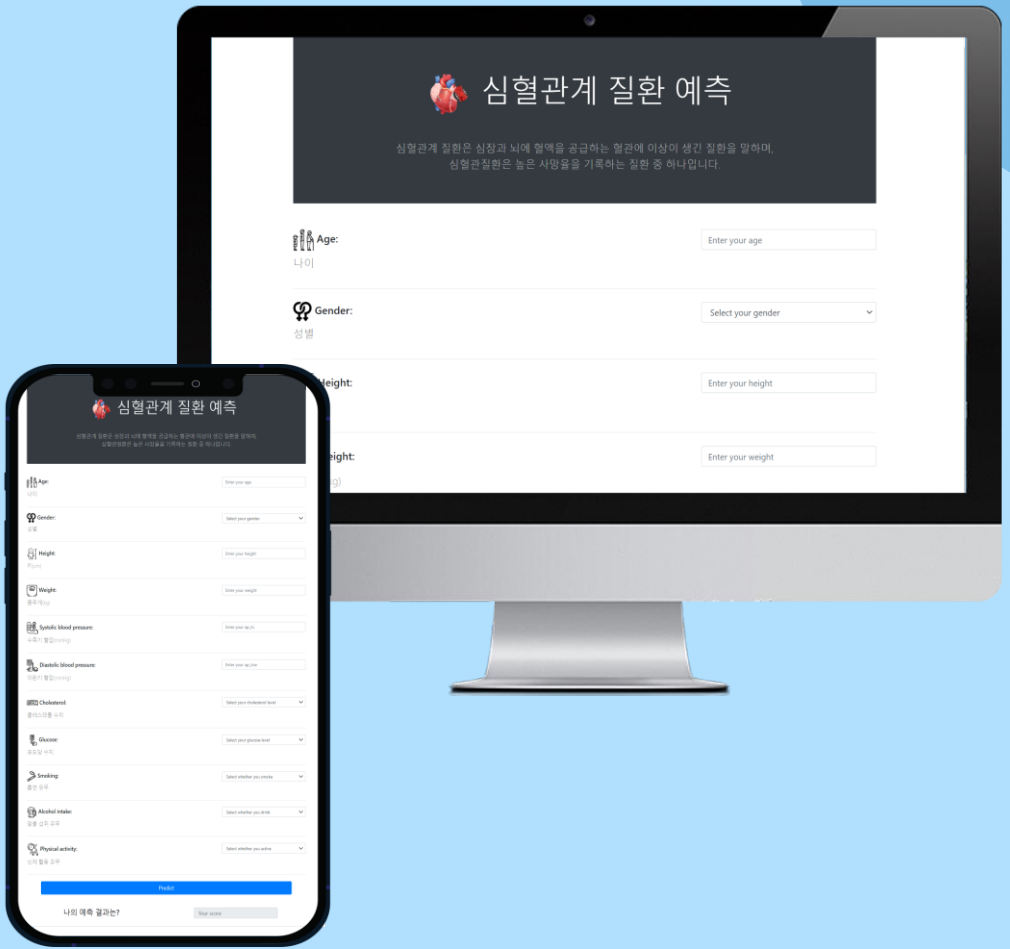
1 프로젝트 소개

심혈관계 질환 예측 서비스

- **핵심 서비스:** 건강 데이터을 활용한 질병 예측 서비스

- **핵심 기술:** Keras, DNN을 이용한 예측 AI

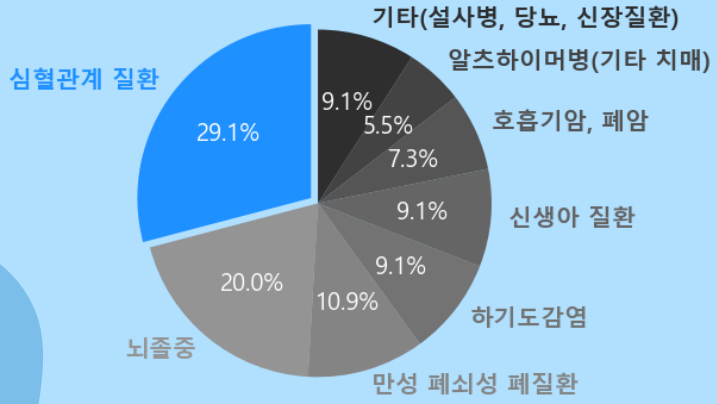
- **서비스 설명:**
  나의 건강 데이터를 바탕으로
  심혈관 질환에 걸릴 확률을 계산해주어
  질병 예방에 도움을 준다.



1 프로젝트 소개

서비스 구현 배경

WHO, 전세계 10대 사망원인



출처: https://www.mediscan.co.kr/kr/customer/card_news.php?bgu=view&idx=1232

전세계 10대 사망원인 중 1위로
매우 높은 사망률을 기록하는 질환.

다양한 심혈관계질환의 원인
→ **현대인들에게 흔히 보임**

1 프로젝트 소개

데이터 소개

Kaggle dataset – Cardiovascular Disease dataset
(https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset)

- **cardio_train.csv**
  70000 rows x 13 columns

- **Features:**
  1. age: 나이
  2. gender: 성별
  3. height: 키
  4. ap-hi: 수축기 혈압
  5. ap-lo: 이완기 혈압
  6. cholesterol: 콜레스테롤
  7. gluc: 포도당
  8. smoke: 흡연
  9. alco: 알코올 섭취
  10. active: 신체활동 유무

- **Target:**
  cardio: 심혈관 질환 유무

결측치, 데이터 분포

```
df.isnull().sum()
```

```
id             0
age            0
gender         0
height         0
weight         0
ap_hi          0
ap_lo          0
cholesterol    0
gluc           0
smoke          0
alco           0
active         0
cardio         0
dtype: int64
```

	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
count	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000
mean	49972.419900	19468.865814	1.349571	164.359229	74.205690	128.817286	96.630414	1.366871	1.226457	0.088129	0.053771	0.803729	0.499700
std	28851.302323	2467.251667	0.476838	8.210126	14.395757	154.011419	188.472530	0.680250	0.572270	0.283484	0.225568	0.397179	0.500003
min	0.000000	10798.000000	1.000000	55.000000	10.000000	-150.000000	-70.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000
25%	25006.750000	17664.000000	1.000000	159.000000	65.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
50%	50001.500000	19703.000000	1.000000	165.000000	72.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
75%	74889.250000	21327.000000	2.000000	170.000000	82.000000	140.000000	90.000000	2.000000	1.000000	0.000000	0.000000	1.000000	1.000000
max	99999.000000	23713.000000	2.000000	250.000000	200.000000	16020.000000	11000.000000	3.000000	3.000000	1.000000	1.000000	1.000000	1.000000

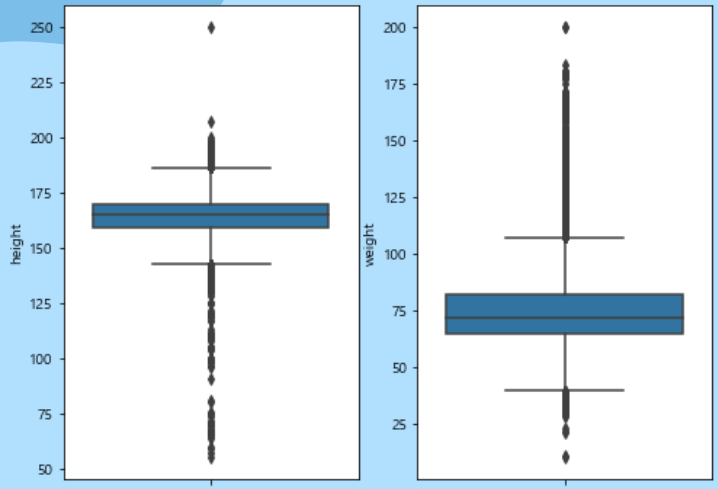
- **결측치:** 존재하지 않음

- **Numerical data:** ‘age’, ‘height’, ‘weight’, ‘ap_hi’, ‘ap_lo’

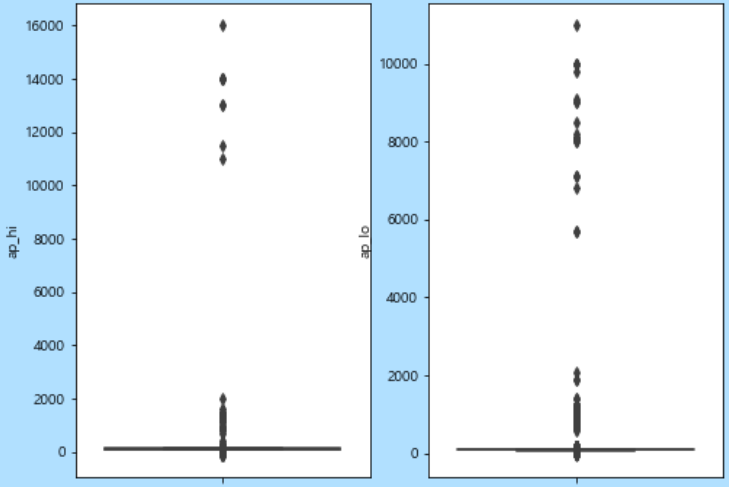
- **Categorical data:** ‘gendet’, ‘cholesterol’, ‘gluc’, ‘smoke’, ‘alco’, ‘active’

2 데이터 전처리

이상치 제거



<height> <weight>



<ap_hi> <ap_lo>

- height: 120이하, 190이상인 데이터 제거
- weight: 30이하, 150이상인 데이터 제거
- ap_hi: 50이하, 200이상인 데이터 제거
- ap_lo: 40이하, 140이상인 데이터 제거

3 EDA

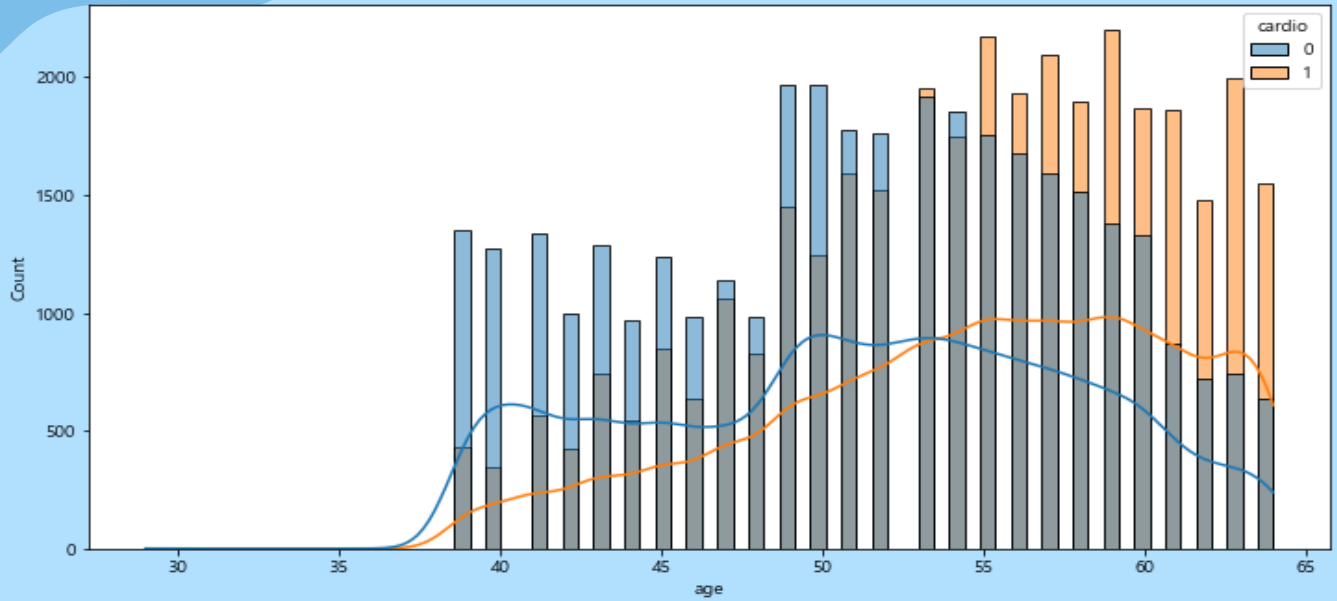
데이터 간 상관관계

- ‘ap_hi’ – ‘ap_lo’ & ‘gender’ – ‘height’ 순으로 상관관계가 높다.

- Target data인 ‘cardio’와 상관관계가 높은 Feature는 ‘age’, ‘ap_hi’, ‘ap_lo’ ‘cholesterol’이다.

3 EDA

‘age’ column

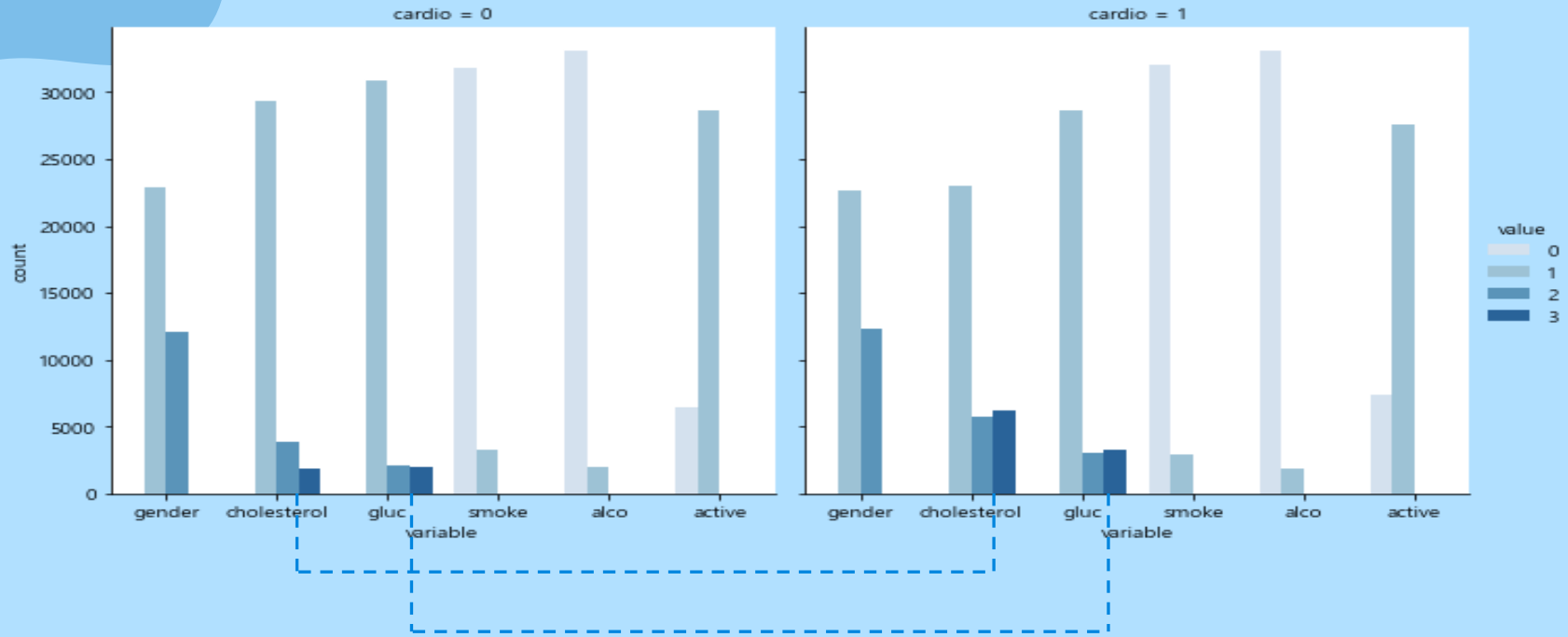


* cardio(1=걸림, 0=안걸림)

- 55세 이상 부터는 심혈관계질환에 걸린 사람이 더 많음을 알 수 있다.

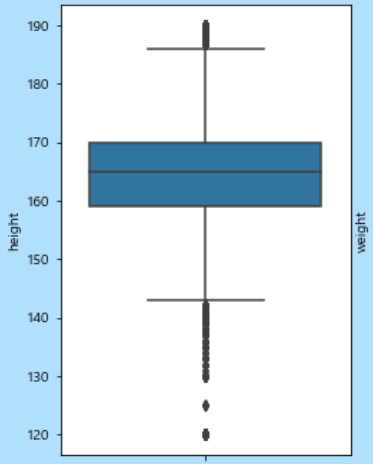
- cardio와 상관관계가 높음을 알 수 있고 예측에 유의미하게 작용할 것이다.

Categorical columns

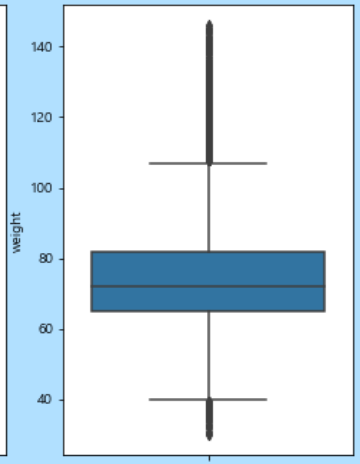


콜레스테롤, 포도당 수치가 좋지 않을 수록
심혈관계 질환에 걸릴 확률이 높다

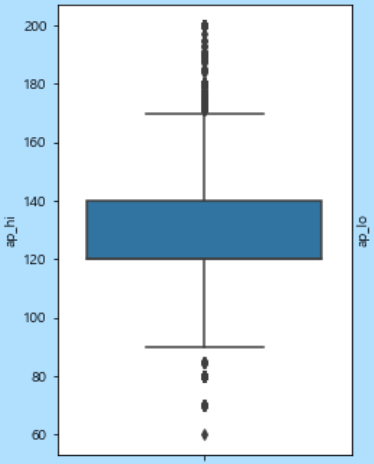
(이상치 처리 후)‘height’, ‘weight’, ‘ap_hi’, ‘ap_lo’



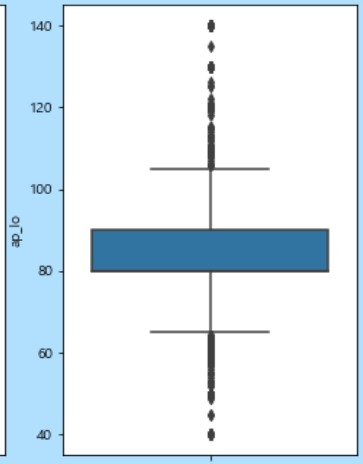
<height>



<weight>



<ap_hi>



<ap_lo>

Base model

<Modeling Flow>

1. Train, Validation, Test split
   -> 6 : 2 : 2

2. Keras 모델 생성(5, 10, activation=linear)

3. Callback함수 구현
   (monitor=val_loss, factor=0.6, patienc=4)

4. Model fit (epochs=100, batch_size=32, shuffle=True)

5. 성능확인

6. Base model을 기준으로 3개의 Case로 테스트해본다.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 5)                 60

 dense_1 (Dense)             (None, 10)                60

 dense_2 (Dense)             (None, 1)                 11

=================================================================
Total params: 131
Trainable params: 131
Non-trainable params: 0
```

< base_model.summary() >

Base model 성능

Loss(cross_entropy): 0.5706
Test accuracy: 0.7218

Case 1 & Case 2

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_6 (Dense)             (None, 2)                 24

 dense_7 (Dense)             (None, 256)               768

 dense_8 (Dense)             (None, 1)                 257

=================================================================
Total params: 1,049
Trainable params: 1,049
Non-trainable params: 0
```

Case1 (2, 256, activation=elu)

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_9 (Dense)             (None, 2)                 24

 dense_10 (Dense)            (None, 512)               1536

 dense_11 (Dense)            (None, 1)                 513

=================================================================
Total params: 2,073
Trainable params: 2,073
Non-trainable params: 0
```

Case2 (2, 512, activation=elu)

Case 1 성능

Loss(cross_entropy): 0.5552(-0.015)
Test accuracy: 0.7301(+0.01)

Case 2 성능

Loss(cross_entropy): 0.5636(-0.01)
Test accuracy: 0.7196(-0.003)

4 데이터모델링

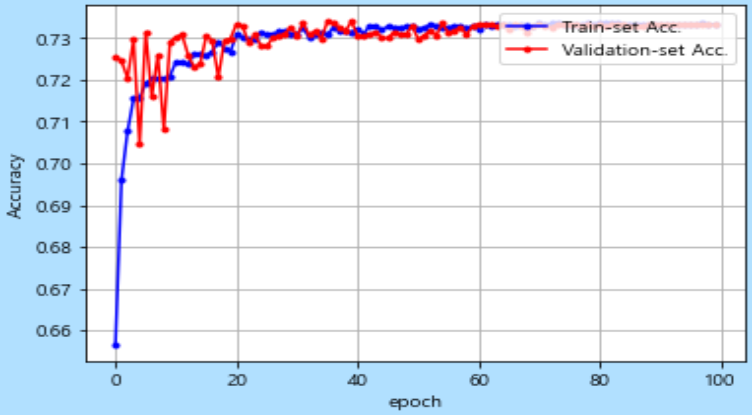
Case 3(Best model)

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_12 (Dense)            (None, 256)               3072

 dense_13 (Dense)            (None, 256)               65792

 dense_14 (Dense)            (None, 1)                 257

=================================================================
Total params: 69,121
Trainable params: 69,121
Non-trainable params: 0
```

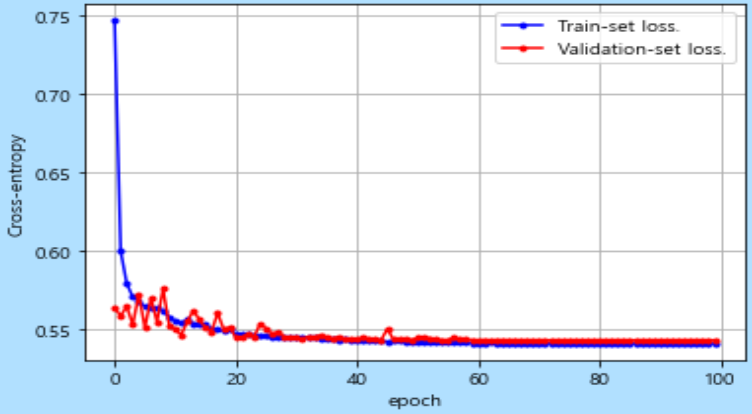
Case3 (256, 256, activation=elu)

Case 3 성능

Loss(cross_entropy): 0.5433(-0.03)
Test accuracy: 0.7332(+0.013)

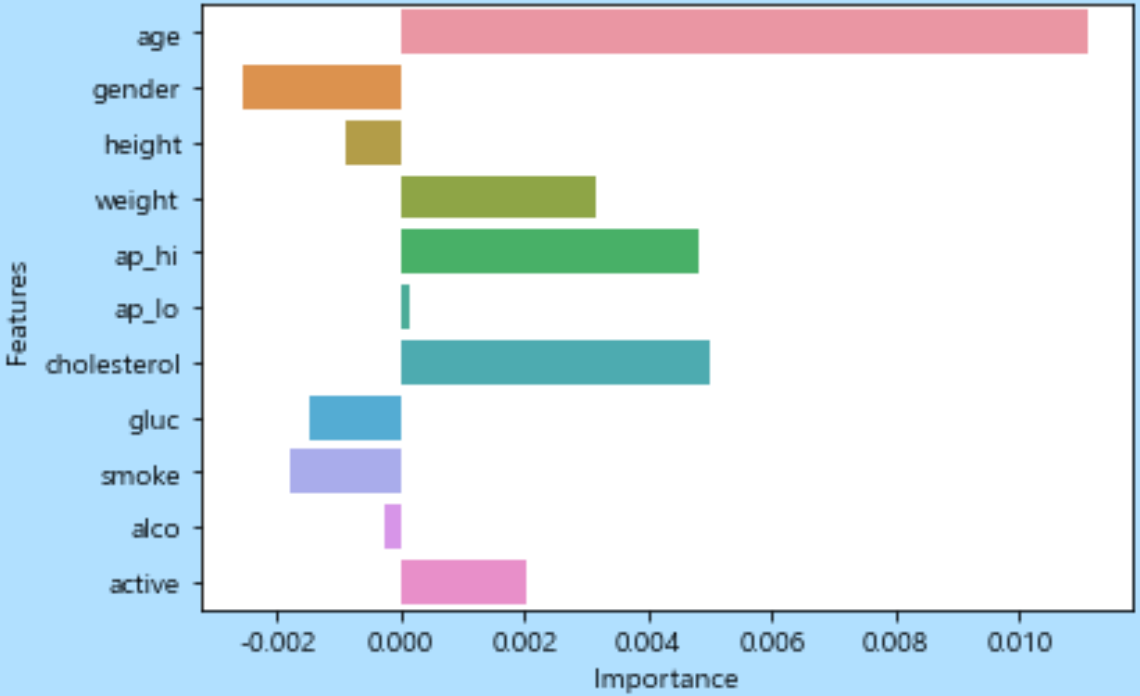


Accuracy history

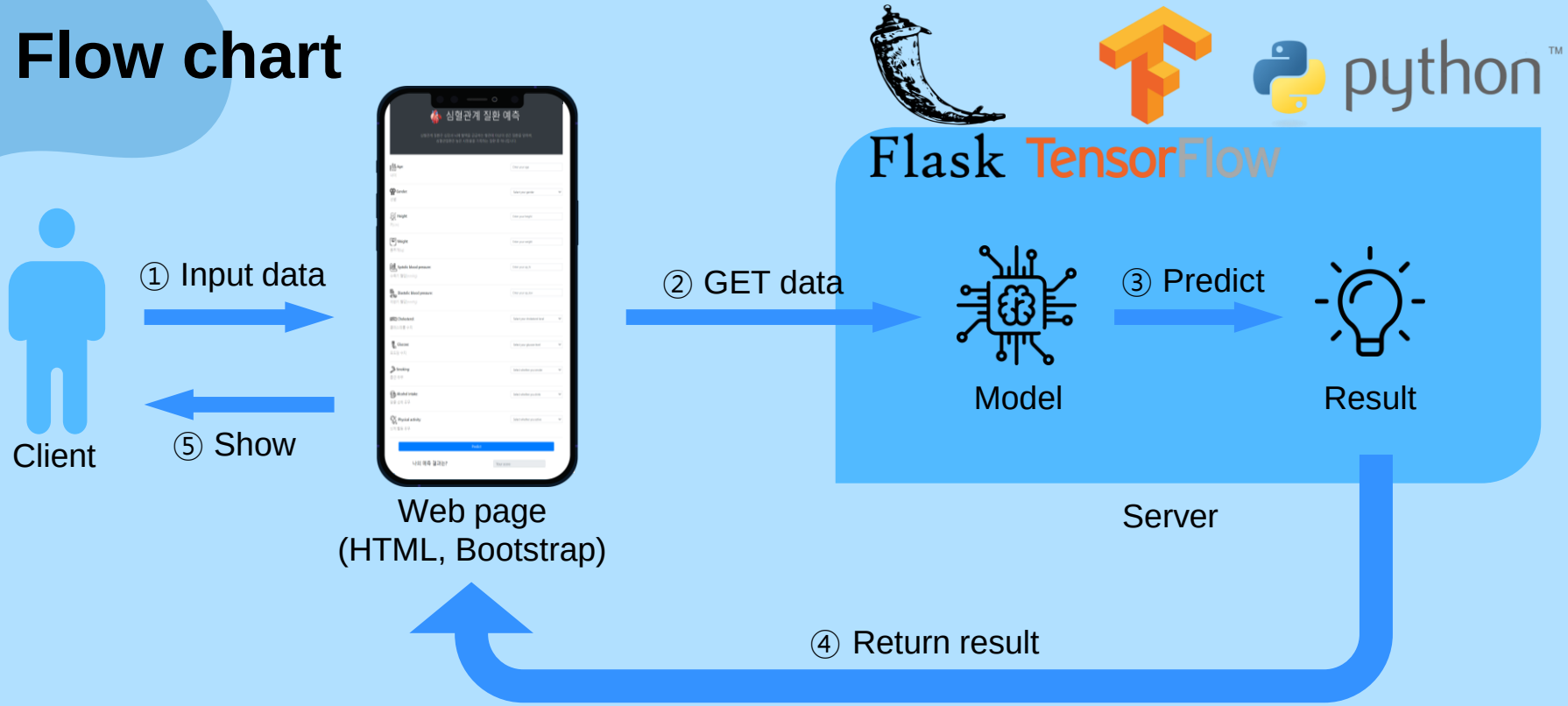


Loss history

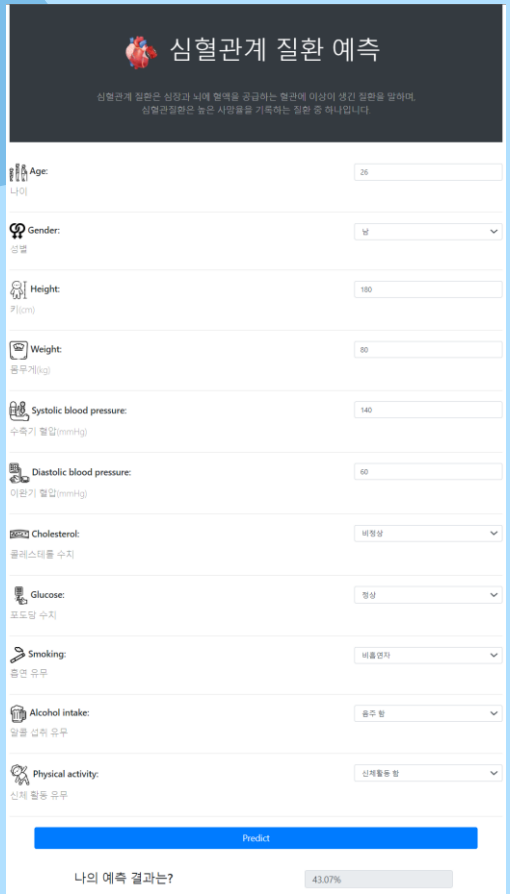
Feature importance

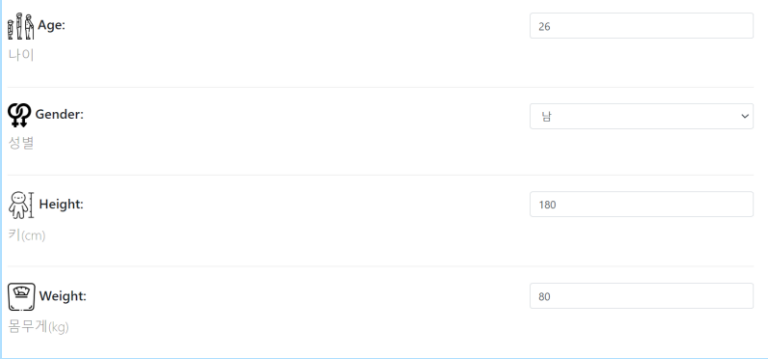


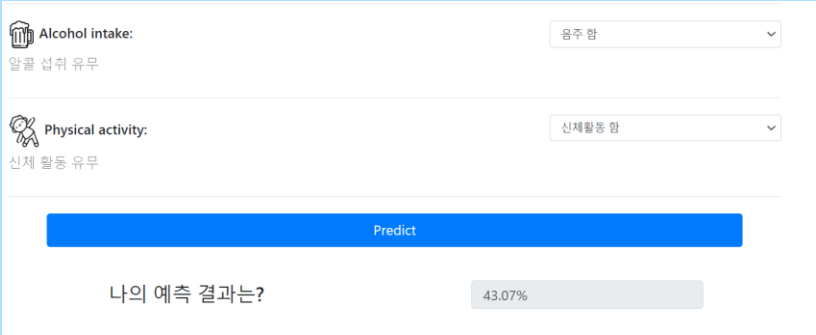
Flow chart


Flask
TensorFlow
python
Client
① Input data
⑤ Show
Web page
(HTML, Bootstrap)
② GET data
Model
③ Predict
Result
Server
④ Return result

5 웹페이지 with Flask







6 개발후기 및 느낀점

Feature engineering의 중요성

구글링도 스킬이 필요

같은 목표를 향해 협업하는 팀원의 필요성

Hyper parameter설정의 어려움

[UX/UI 보완]
* 질병 예방법에 대한 내용추가
* 모바일 환경에 맞게 크기 조절

[모델링 보완]
* 충분한 hyper parameter tuning
* 다양한 Activation함수 이용

Thank you