

AI 부유물 탐지 시스템 구축 프로젝트

개인 프로젝트

조원: 백동현

목차

1. 의도
2. 과정
3. 시행착오와 개선
4. 결과와 후기

1. 의도

- 2개월 간 파이썬과 기초적인 ai 이론, roboflow같은 툴의 조작을 배우며 공부했던 내용들을 이번 프로젝트에 총집하였습니다
- 그간 배웠던 내용들을 정리하고 응용해,특정데이터에 내가 생각하는 유의미한 자료를 선별해 시각화시키는것.
- 배웠던 내용들이 왜 필요한지 직접 복습해 보면서 체득하고, 어디에 사용할 수 있는지 직접 생각해 보는 것

2. 과정

- 파이썬에서 주어진 데이터를 딥러닝해 기본적인 계산과 결과를 분석하는 과정을 첫 번째로 넣었습니다
- 그 다음으로 계산된 결과들을 분류해 표를 만들었고, 그걸 시각화 해 유의미한 정보로 가공하는 과정을 두 번째로 했습니다.
- 세 번째로는 roboflow같은 툴들을 직접 이용해 보면서, 기초적인 ai이론들이 어떻게 이용되는지, 통계나 분류같은 방법으로 감을 익혔습니다.
- 마지막으로 tts를 파이썬내부에서 실행하지 않고 자바html에 연동해서 실행시켜보았습니다.


- main

Title

localhost:1001

Home ML/DL 시각화 RoboFlow text 음성

1. tts 입력
2. tts 출력



2025년 9월

today < >

일	월	화	수	목	금	토
31일	1일	2일	3일	4일	5일	6일
7일	8일	9일	10일	11일	12일	13일
14일	15일	16일	17일	18일	19일	20일
21일	22일	23일	24일	25일	26일	27일
28일	29일	30일	1일	2일	3일	4일
5일	6일	7일	8일	9일	10일	11일

오늘 날씨

온도: 24.5°C

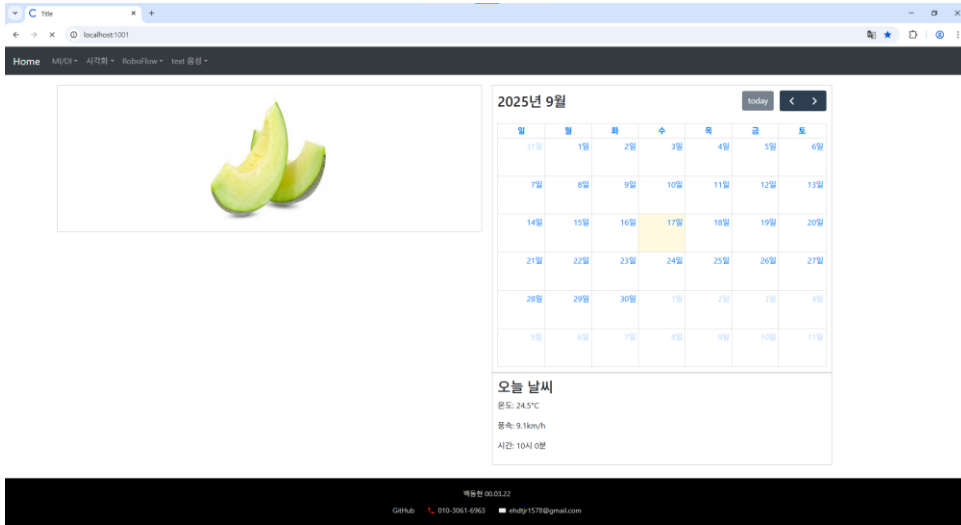
풍속: 9.1km/h

시간: 10시 0분

백동현 00.03.22

GitHub 010-3061-6963 ehdtr1578@gmail.com

- main_menu1



Iris, Indians, wine 데이터들을 딥러닝 하는 과정입니다.
데이터를 딥러닝 해 정확도와 손실률을 구했습니다.
이후 파이썬에서 구해진 정확도와 손실률을 JSON으로 HTML에 띄워주고,
파이썬에서 계산된 값을 txt 파일로 저장했습니다.

- main_menu1

```
indians.py - C:\AI\python\project\chart\indians.py (3.10.11)
File Edit Format Run Options Window Help

# 손실률, 정확도 스프링 부트로 보내기
result={
    "seq":None,
    "loss":ss[0],
    "accuracy":ss[1]
}

url = "http://localhost:1001/python/indianssave"
requests.post(url, json=result)

# -----

# 구조 json 파일 저장
save_dir = r"C:\AI\python\project\chart\indians"
json_path = os.path.join(save_dir, "indians.sequential.json")
weights = os.path.join(save_dir, "indians.weights.h5")
evaluate_path = os.path.join(save_dir, "indians.evaluate.json")

sequential = model.to_json()
with open(json_path, "w") as json_file:
    json_file.write(sequential)
print("...indians.sequential.json 저장 완료")

# 가중치 h5 파일 저장
model.save_weights(weights)
print("...indians.weights.h5 저장 완료")

# evaluate json 파일 저장
evaluate = {
    "loss": ss[0],
    "accuracy": ss[1]
}
with open(evaluate_path, "w", encoding='utf-8') as f:
    json.dump(evaluate, f, ensure_ascii=False, indent=4)
print("...indians.evaluate.json 저장 완료")
```

```
indians.evaluate - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

{
  "loss": 0.8109086155891418,
  "accuracy": 0.51953125
}
```

내 PC > 로컬 디스크 (C:) > AI > python > project > chart > indians		
이름	수정한 날짜	유형
indians.evaluate	2025-09-17 오후 8:17	JSON 파일
indians	2025-09-16 오후 5:59	Python File
indians.sequential	2025-09-17 오후 8:17	JSON 파일
indians.weights	2025-09-17 오후 8:17	H5 파일
indiansplt1	2025-09-16 오전 10:08	Python File
indiansplt2	2025-09-16 오전 10:08	Python File
indiansplt3	2025-09-16 오전 10:08	Python File

내 PC > 로컬 디스크 (C:) > AI > python > project > chart > iris		
이름	수정한 날짜	유형
iris.evaluate	2025-09-17 오후 8:16	JSON 파일
iris	2025-09-17 오전 10:21	Python File
iris.sequential	2025-09-17 오후 8:16	JSON 파일
iris.weights	2025-09-17 오후 8:16	H5 파일
irisplt	2025-09-16 오전 10:07	Python File

내 PC > 로컬 디스크 (C:) > AI > python > project > chart > wine		
이름	수정한 날짜	유형
wine.evaluate	2025-09-17 오후 8:18	JSON 파일
wine	2025-09-17 오전 10:22	Python File
wine.sequential	2025-09-17 오후 8:18	JSON 파일
wine.weights	2025-09-17 오후 8:18	H5 파일
wineplt1	2025-09-16 오전 10:08	Python File
wineplt2	2025-09-16 오전 10:08	Python File

- main_menu1

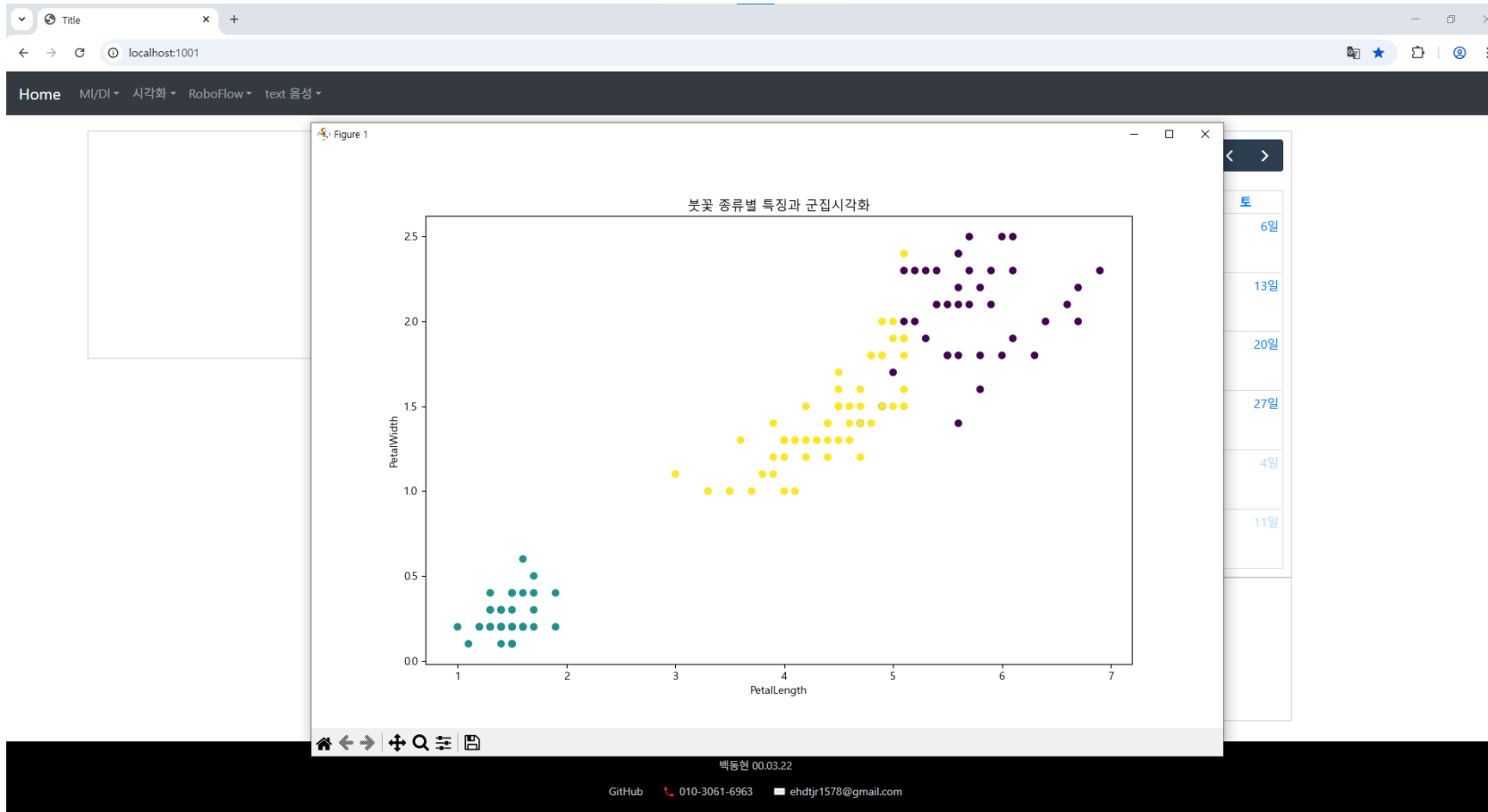
딥러닝 결과			
전체 보기 붓꽃 데이터 인디언 당뇨 데이터 와인 데이터			
번호	손실률	정확도	기능
3000	0.22656631469726562	0.9247229099273682	삭제

백동현 00.03.22

GitHub 010-3061-6963 ehdtjr1578@gmail.com

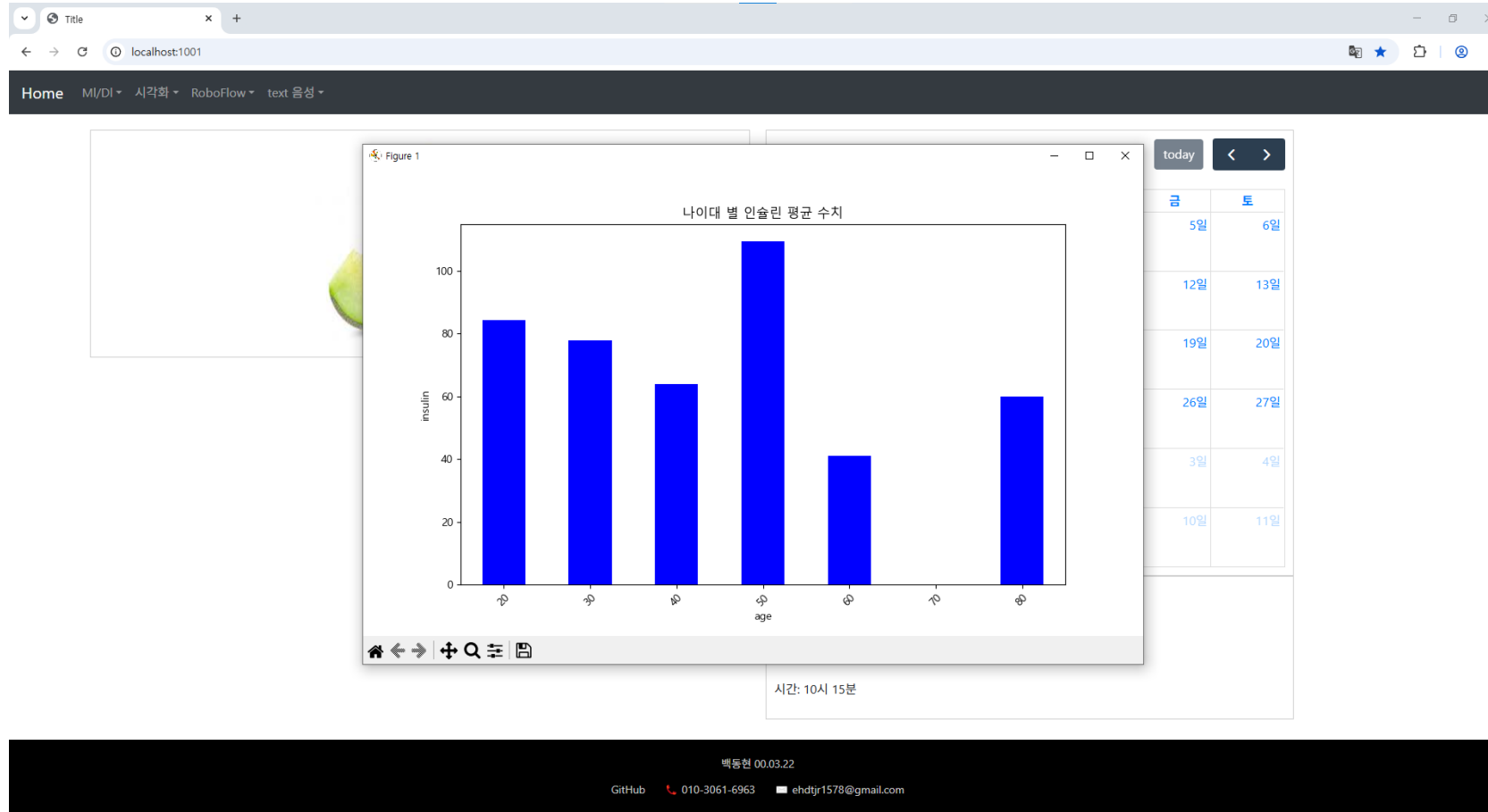
나온 손실률과 정확도를 out.html로 빼두었고,
시퀀스로 자동 넘버를 부여해, 필터로 구분해 주었습니다.

- main_menu1



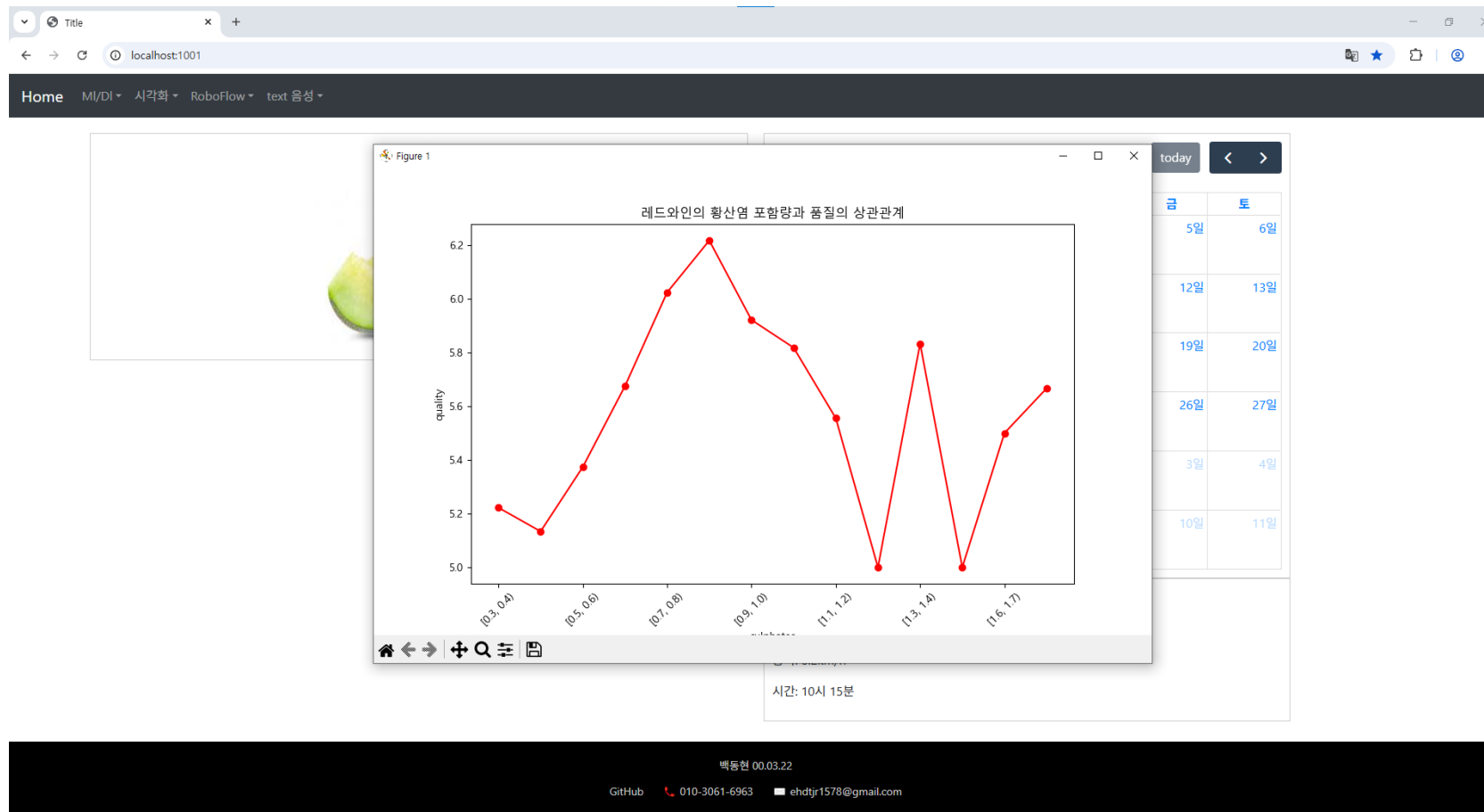
K계열 군집화를 사용해 붓꽃 데이터를 분류해보았습니다.
원 핫 인코딩으로 3개의 결과를 전처리 한 후 각 값들을
다른 색으로 표시해, 꽃의 크기와 모양을 기준으로 시각화하였습니다.

- main_menu1



막대 그래프를 이용해 파마 인디언 부족들의 당뇨병 진단률을 시각화 했습니다.
확진자의 특징을 알아보기 위해, 우선 나이를 기준으로 확진자를 알았고,
연관성을 알아보기 위해 연령 별 비만률과, 인슐린 수치를 알아보았습니다.

- main_menu2



와인은 이진분류가 가능한데, 실물 이미지 없이 데이터만 보았을 때 어떤 특정 값으로
구분이 가능할까 라는 의문을 주제로 시각화 하였습니다.
그 후에 품질을 기준으로 와인의 구분을 특정하는 휘발성 산도의 평균 값을 대입해 시각화했습니다

- main_menu3

Pose Estimation

localhost:1001/pose

Home ML/DL 시각화 RoboFlow text 음성

포즈 추정 테스트

분석에 시간이 조금 소요됩니다..

파일 선택 jump.jpg 분석 시작

```
{
  "status": "success",
  "message": "포즈 추정이 완료되었습니다.",
  "landmarks": [
    {
      "오른쪽 어깨 x좌표": 0.42011305689811707,
      "오른쪽 어깨 y좌표": 0.2183251678943634,
      "오른쪽 골반 x좌표": 0.4543652832588087,
      "오른쪽 골반 y좌표": 0.49085164070129395
    }
  ]
}
```

돌아가기

백동현 00.03.22

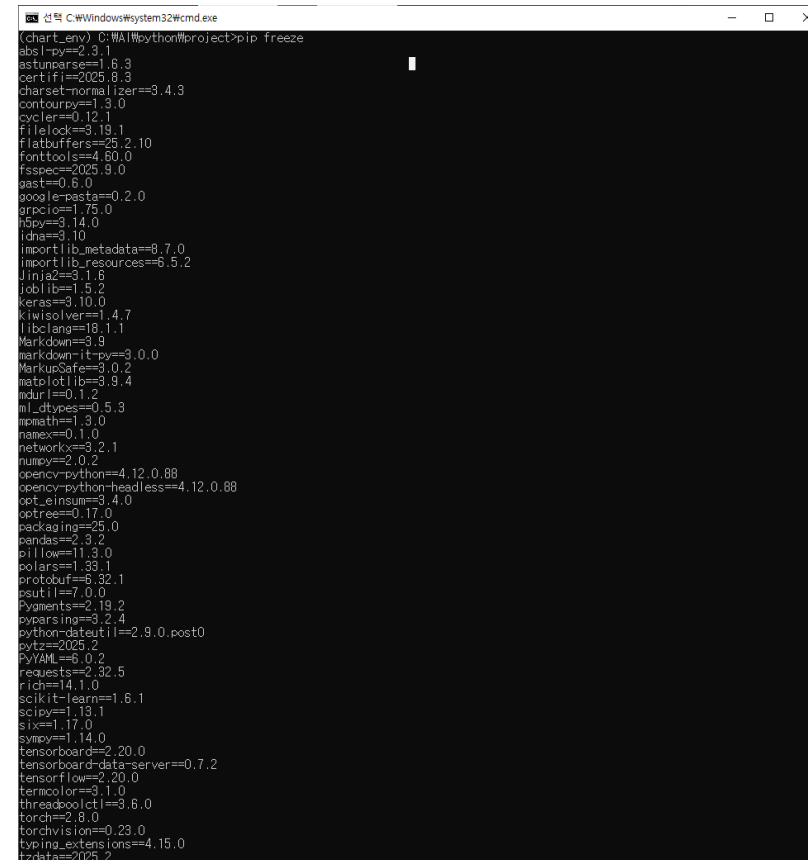
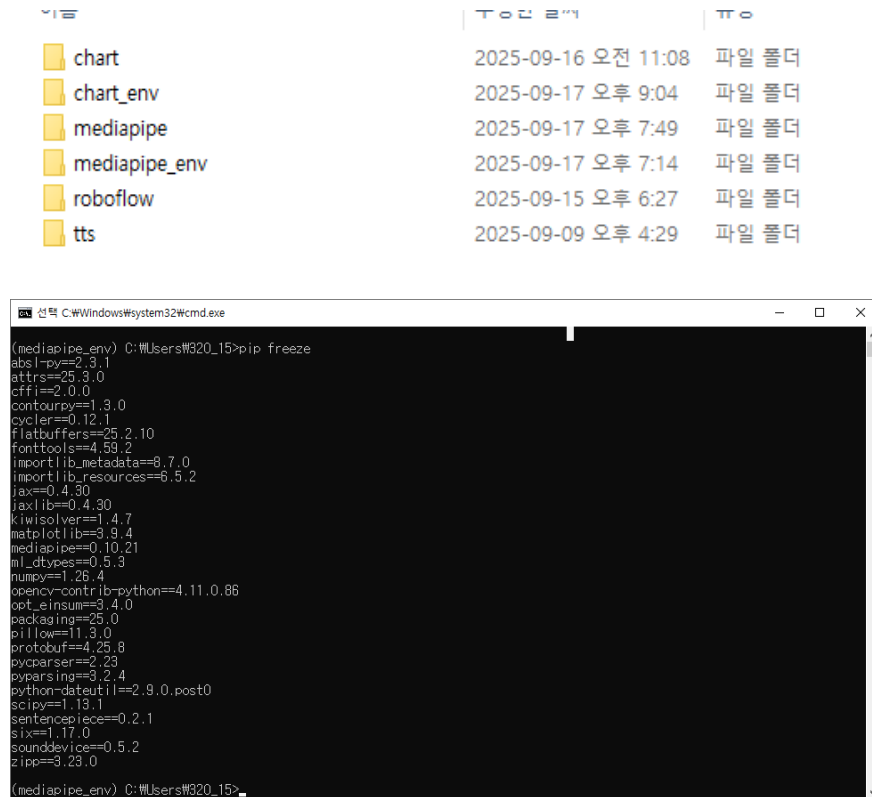
GitHub 010-3061-6963 ehdtjr1578@gmail.com



Jump.jpg

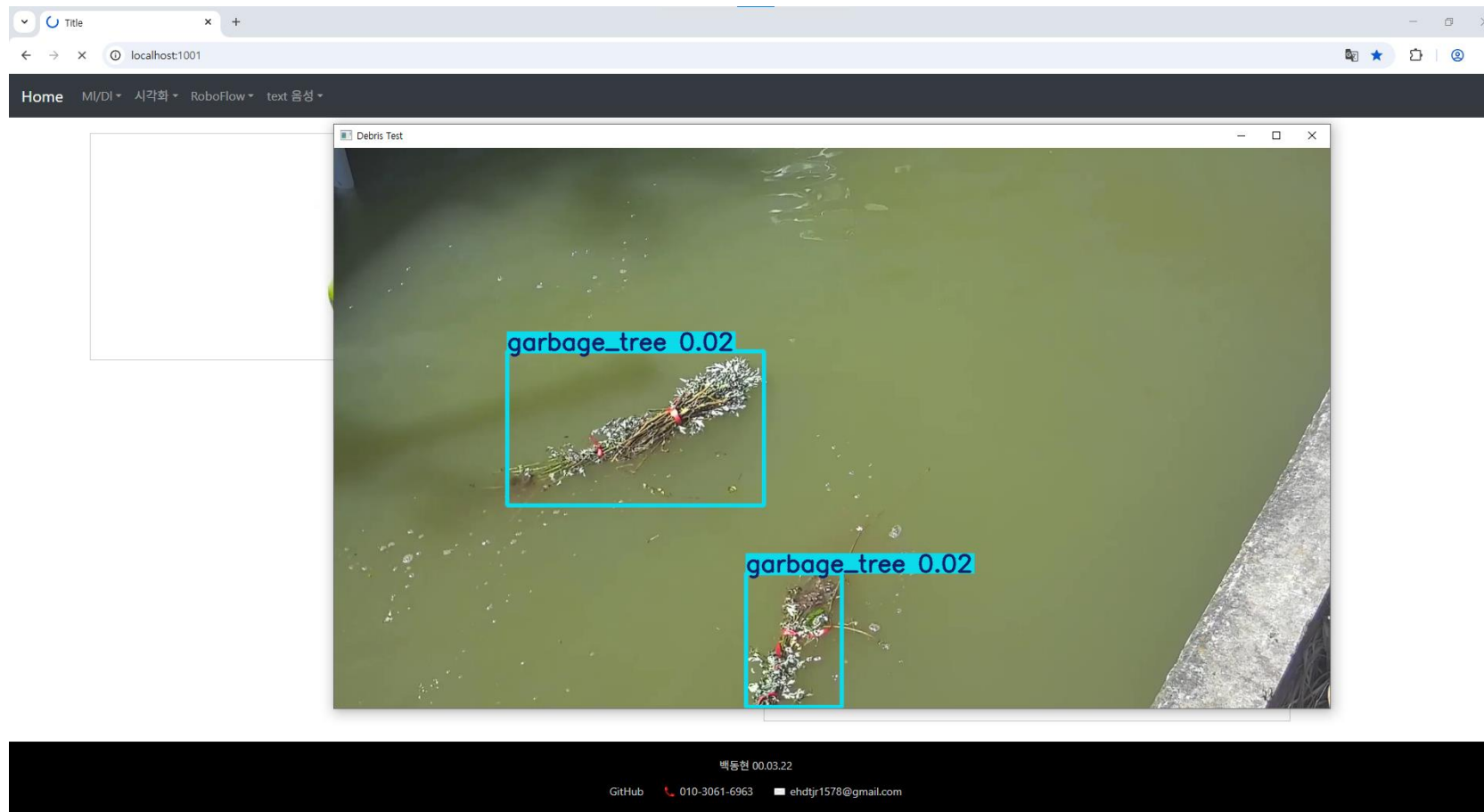
다음은 google media pipe api를 이용한 포즈 추정 테스트 입니다.
사진 속 인물의 골격을 api가 자동으로 인식하고,
사진 상의 골격 좌표를 추적해 줍니다.

- main_menu3



처음의 의도는 roboflow는 할 줄 아니 다른 api를 사용해 보고 싶다는 점이었습니다.
그런 중에 딥러닝과 google mediapipe의 tensorflow 라이브러리가 충돌이 있다는 지점을 발견했습니다.
그래서 가상환경을 주어 구버전 tensorflow와 mediapipe 라이브러리를 설정해 따로
환경을 마련해 주었습니다.

- main_menu3



Roboflow를 이용한 움직임 감지 프로그램 입니다.
-강가 부유물 탐지-

- main_menu3

Debris Test

ocean_fish 0.39

ocean_fish 0.08

ocean_fish 0.45

ocean_fish 0.37

ocean_fish 0.27

ocean_fish 0.16

ocean_fish 0.04

today < >

목	금	토
4일	5일	6일
11일	12일	13일
18일	19일	20일
25일	26일	27일
2일	3일	4일
9일	10일	11일

오늘 날씨

온도: 24.6°C

풍속: 7.1km/h

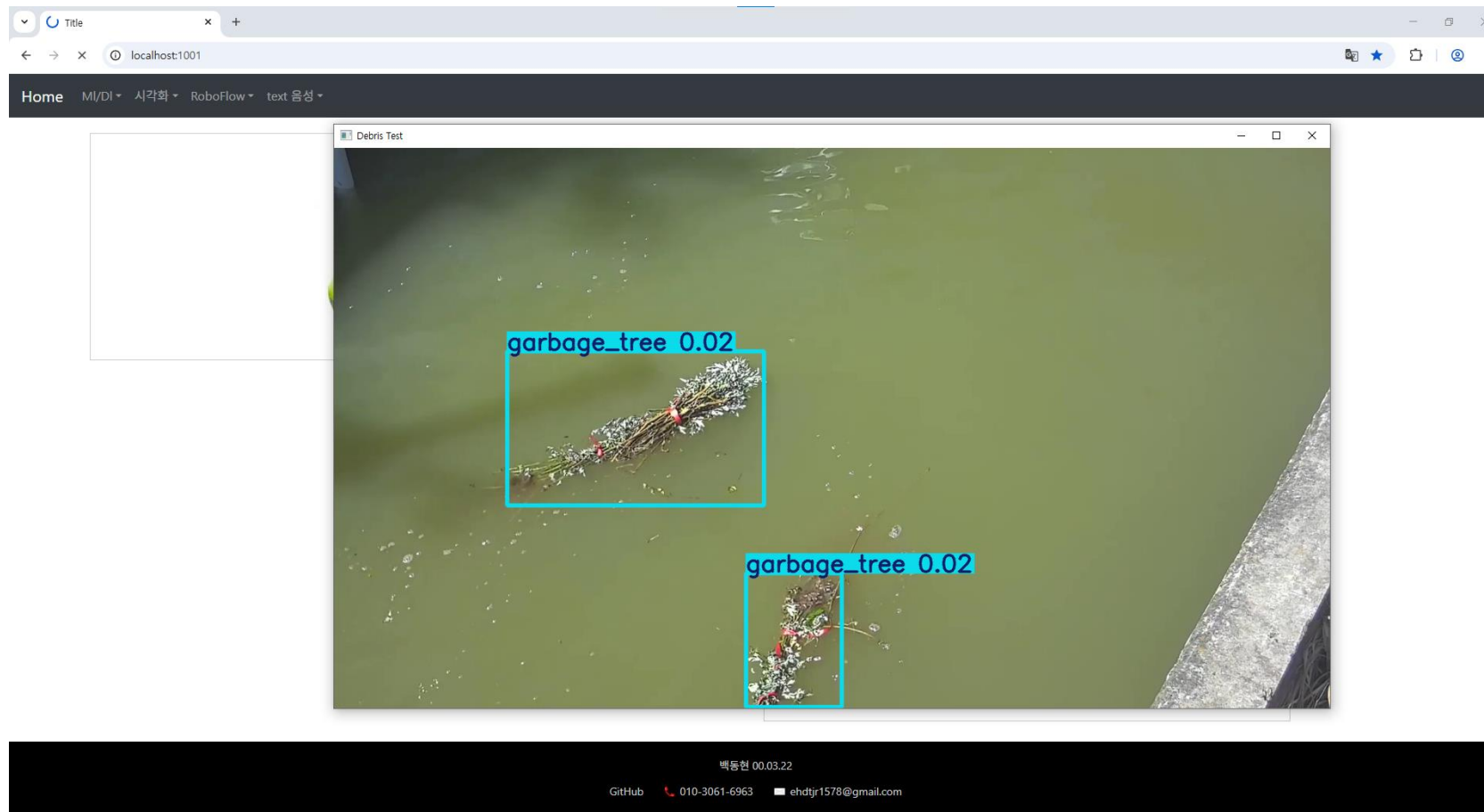
시간: 10시 30분

백동현 00.03.22

GitHub 010-3061-6963 ehdtjr1578@gmail.com

-해양 생물 탐지-

- main_menu3




-교차로 보행자 인식

- main_menu3

Web browser interface showing a main menu and a video feed with object detection results.

Navigation bar: Home MI/DI 시각화 RoboFlow text 음성

Video Feed (Debris Test):



Calendar (Today):

화	수	목	금	토		
2일	3일	4일	5일	6일		
9일	10일	11일	12일	13일		
16일	17일	18일	19일	20일		
23일	24일	25일	26일	27일		
30일	1일	2일	3일	4일		
5일	6일	7일	8일	9일	10일	11일

오늘 날씨

온도: 24.6°C

풍속: 7.1km/h

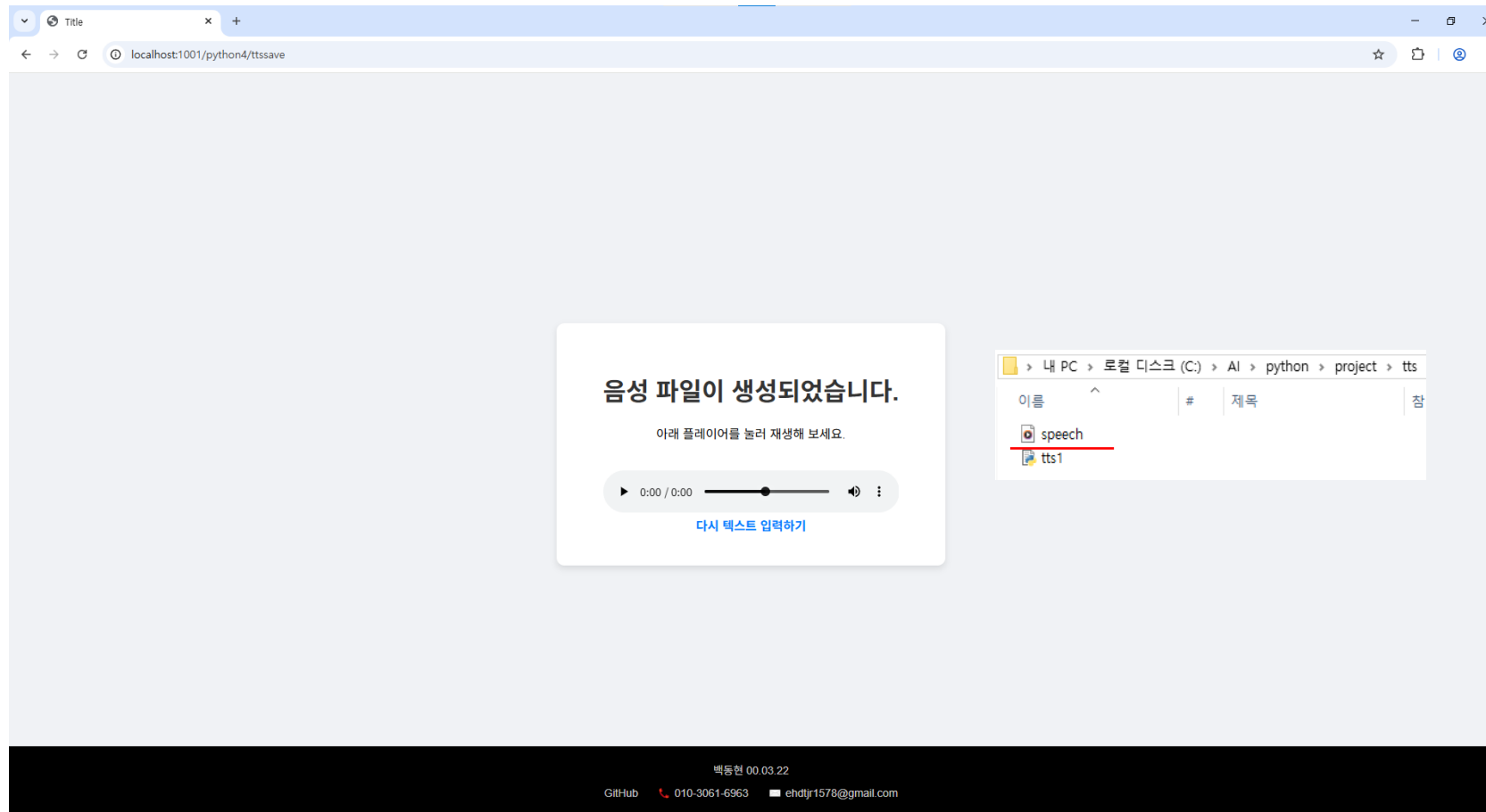
시간: 10시 30분

백동현 00.03.22

GitHub 010-3061-6963 ehdtjr1578@gmail.com

-신호등 교차로 보행자 인식-

- main_menu3



Text to speech 입니다.
가나다 를 입력하면 텍스트를 mp4 파일로 변환해 파일에 저장해주고,
저장된 mp4파일을 자바 html에서 직접 읽어주는 방식입니다.

3. 시행착오와 개선

- 가장 기억에 남는 오류는 라이브러리 충돌 문제였습니다.

파이썬의 버전에 맞춰 라이브러리가 깔려 충돌이 일어나는 줄 알았습니다.

그래서 3.9, 3.10, 3.13 등 버전마다 깔아서 재설치해보고, 환경변수도 계속 고쳐봤지만 효과가 없었습니다.

알고보니 파이썬의 버전이 문제가 아니라 mediapipe api에서 구버전 tensorflow(protobuf~3.9) 를 기반으로 구동된다는걸 알게됐습니다.

가상환경은 배운 내용이 아니었기에 끝까지 미루고 있던 방법이었는데 답이 없어서 막상 해보니 그렇게 어렵지 않았고, 이번 프로젝트 중 가장 기억에 남는 경험이었다고 생각합니다.

3. 시행착오와 개선

RestController와 controller의 차이

RestController는 그 자체로 쓰임새가 정해진 어노테이션이라는 생각으로 쪽 작업을 진행 하고 있었습니다.

첫 번째 작업에서 3개의 데이터에 대한 딥러닝이 끝나면 손실률과 정확도를 json을 통해 html로 받으려는 작업을 한 controller에서 받으려고 했습니다.

그런데 이 작업 중에 저장된 값을 삭제하는 기능을 넣으려다 restcontroller는 html을 반환하지 못한다는게 떠올랐습니다.

이때 제미나이의 도움으로 restcontroller를 주지 않고 각 메서드에 responsebody어노테이션을 할당해주면 된다는걸 알았습니다.

3. 시행착오와 개선

이 ppt를 완성하고 git에 ppt를 올리려다 실수로 파일을 백업하지 않고 git reset - hard 해버렸습니다.

덕분에 오늘까지 기한인데 pdf로 백업해 둔 ppt를 보며 다시 만드는 사고가 있었습니다.

오늘 일로 git에 무언가를 올릴 때는 반드시 백업을 더블체크 해야겠다고 굳게 다짐했습니다.

4. 후기

감사합니다.