코딩과정 보고서

이런 내용은 들어갔으면 좋겠다 싶은 것, 그리고 밑부분은 내가 한 것들 적은 거

근데 밑부분에 쓴거가 뒤로 갈 수록 말이 안 매끄럽고 이해 안 가게 쓴 부분들 엄청많으니까 그냥 복붙하면 안될거야

보고서 목차

1. 서론

- 문제 의식 (잔반이 많다)

- 프로젝트 개괄 (아이들의 식판 사진 수집한 다음 구글 티쳐블 머신을 이용한 이미지 분석으로 결과를 내야겠다)

2. 과정

- 조사 과정

- 인공지능 이미지 분석 및 크롤링을 통한 자동화 과정(이 부분 뭐적어야할지 밑에 쓸거얌)

3. 결과

4. 실천

- 급식소에 보고서 전달

5. 소감

3줄 요약

1. 지금 github에 내가 코딩 과정 원리랑 보고서(사회용) 쓸 계획 대충 써서 올려놓은 것 있음

2. 이것 보고 역할 분담 다시해야할 듯

3. 결과랑 실천 부분만 빼고 다 쓸 수 있을 것 같아.

1. 자료 정리

모인 사진을 우리 모둠의 github 저장소를 만들어 올렸고, 같이 협업했다.

정리하고 가공하니 1000여개정도의 사진이 모였음

* 자료가 방대해 모종의 빅데이터 분석 기술을 사용할 수 밖에 없음

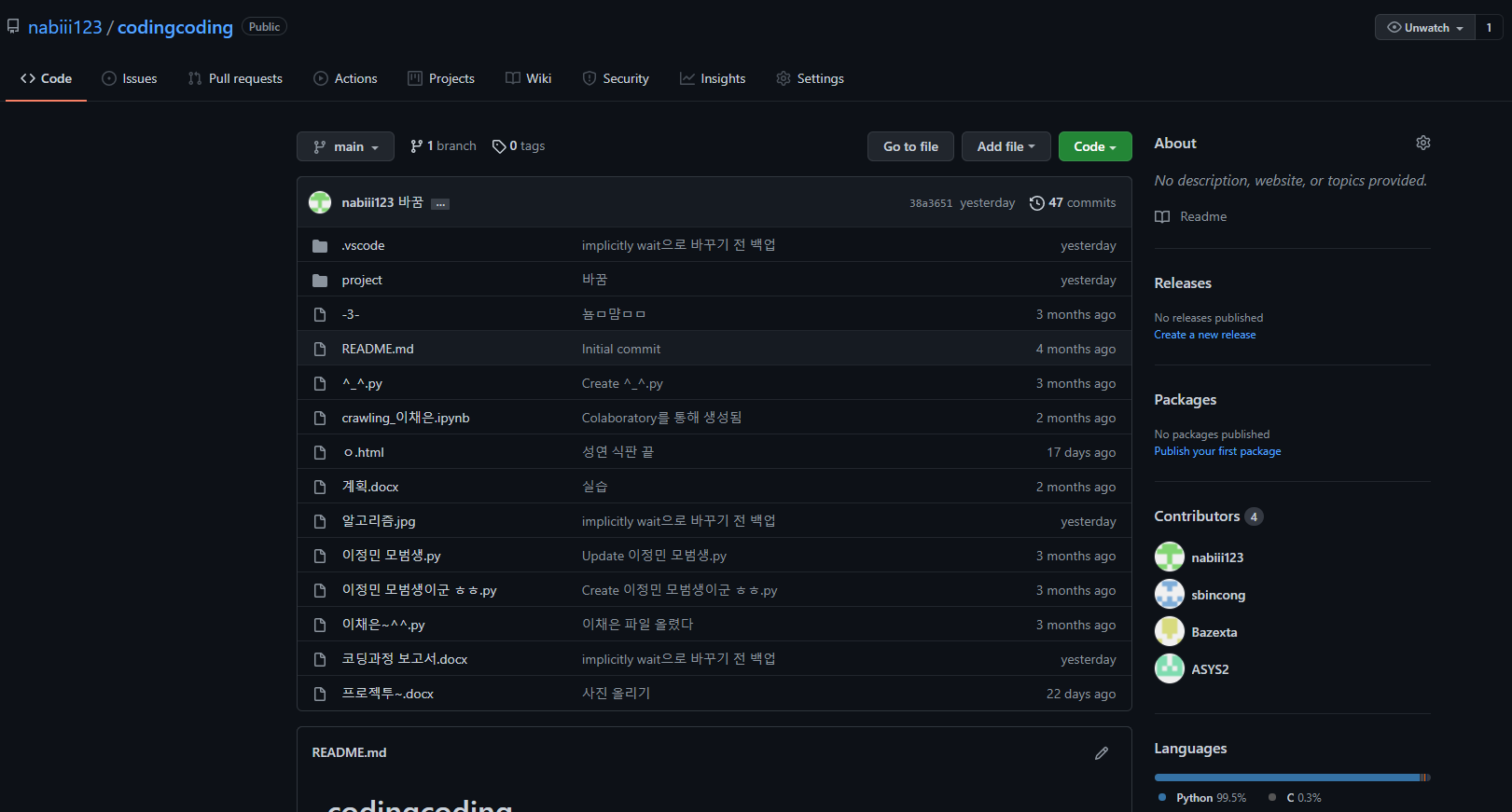
Figure

이때, github가 뭐하는 곳인지 간단 설명 고고 (다양한 기능 있지만 우리의 경험 그대로 쓰면 될 듯)

이 주소는 우리 모둠의 github 저장소 여기에 우리 피땀눈물 다있으니까 봐달라고 해야해

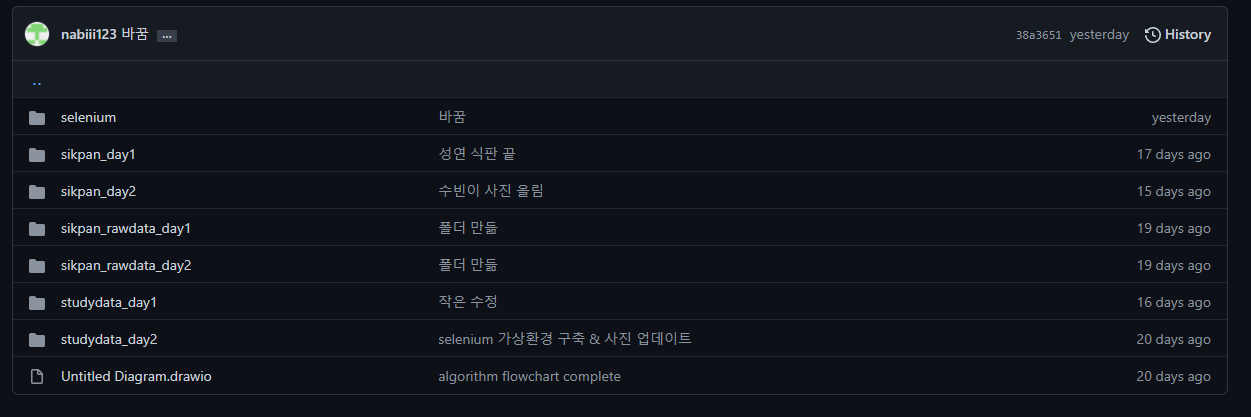
<https://github.com/nabiii123/codingcoding>

2. 우리 모둠 github 저장소 둘러보기



Figure

왼쪽 하단에 있는 아이디들은 우리 모둠원들이다. 중간에 있는 잡다한 파일 중 project에 들어가 보면..



Figure

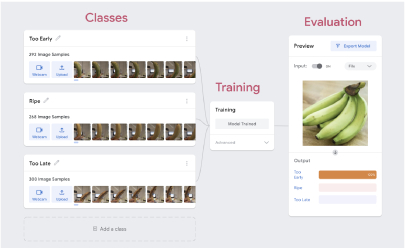
Selenium-> 프로그램 들어있음 (dataprocess.py가 해당 파일)

Sikpan\_ -> 학생들 식판

Sikpan\_rawdata -> 가공 안된 학생들 식판 자료

Studydata\_ -> 인공지능 학습시킬 데이터 (특히 이거 꼭 봐주셨음 좋겠다)

2. 코딩 과정 설명

1. 인공지능 사진 분석

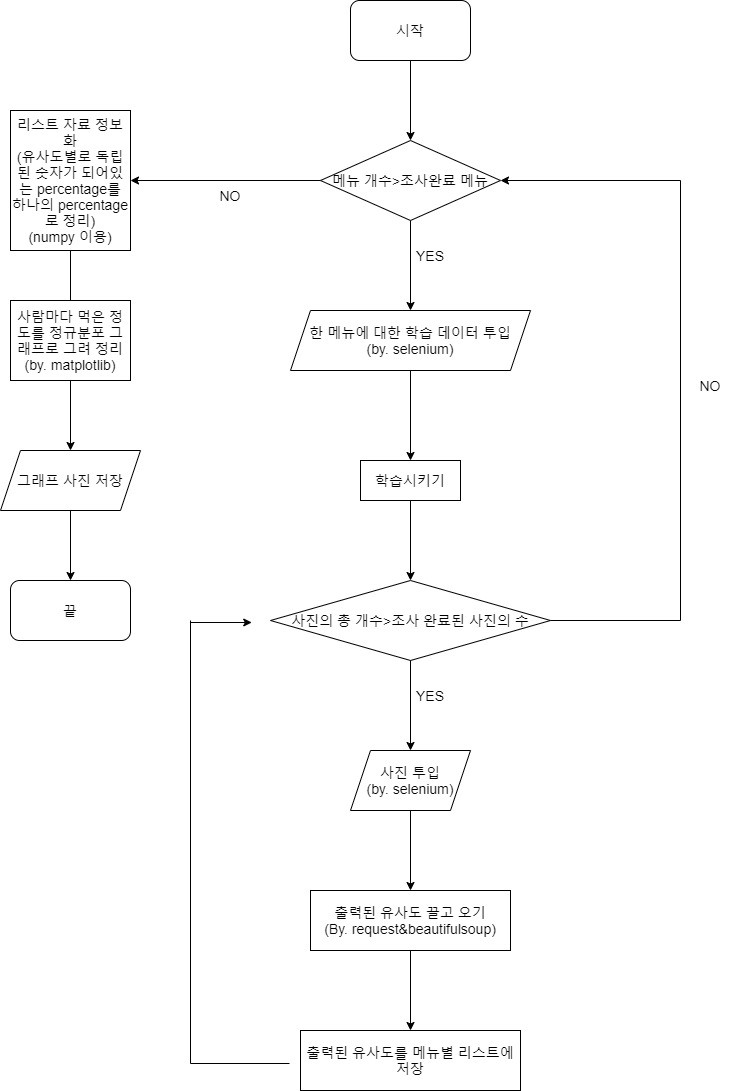
Figure

우리는 구글에서 무료로 제공하는 인공지능 모델인 ‘구글 티쳐블 머신’ 중 사진 분석 기능을 이용할 것이다.

옆에 보이는 것과 같이 다량의 사진들로(사진에서 classes부분) 인공지능을 학습시키고, 다른 사진을 넣어 결과를 알 수 있는 서비스이다.

2. 크롤링을 이용한 전 과정 자동화

우리가 분석해야할 사진은 600여개 정도인데, 우리가 사용하려는 인공지능 모델인 ‘구글 티쳐블 머신’은 사진을 하나하나 투입해서 하나하나 결과가 나오는 시스템이다. (위 사진 참조) 그런데, 우리가 사진을 하나하나 손수 투입하고, 결과를 하나하나 엑셀파일에 받아 적는 것은 손이 무척 많이 가는 일로, 인공지능과 결합해 나름의 빅데이터 정보처리를 하겠다는 우리의 목적과 모순되는 일이다.

그래서, 우리는 파이썬으로 ‘크롤링’(웹을 자동으로 조작하고, 정보를 끌어오는 기술을 통칭하는 말)을 해서 파일을 투입하고, 분석결과를 끌어오고, 그 결과를 컴퓨터에 저장하고 정규분포 그래프로 그리는 것까지 자동으로 해주는 프로그램을 짜야겠다고 생각했다. 그래서 아래와 같이 알고리즘을 구상했다.

Figure

3. 크롤링 코딩

‘크롤링’을 하기 위해서 나는 ‘셀레니움’이라는 파이썬의 모듈을 이용했다. 이때, 모듈이란 사람들이 많이 쓰는 함수 같은 것을 모아놓아 쓰기 쉽게 해놓은 것을 말한다. 이 보고서는 문제 해결이 중점이 되므로 기술적인 설명은 위 알고리즘 플로우 차트를 참고하고 크게 하지 않겠다.

아까 써놓은 github저장소의 selenium/dataprocess.py파일에 들어가면 원문을 볼 수 있다. 참고로 셀레니움 모듈 외의 오픈소스는 활용하지 않았고 우리가 모두 직접 쓴 코드이다.

[코드 전문]

import selenium

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import Select

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

import os

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#element = wait.until(EC.element\_to\_be\_clickable((By.ID, 'someid')))

# wait = WebDriverWait(driver, 10)

URL = 'https://teachablemachine.withgoogle.com/train/image'

options = webdriver.ChromeOptions()

options.add\_experimental\_option("excludeSwitches", ["enable-logging"])

driver = webdriver.Chrome(options=options)

driver.implicitly\_wait(10) # seconds

menu={

    "day1":["egg","kimchi","namul","rice","soup"],

    "day2":["hobak","kimchi","potato","rice","soup"]

    }

menu\_result={

    "day1":[[],[],[],[],[]],

    "day2":[[],[],[],[],[]]

    }

for key in menu:

    menukey=0

    for food in menu[key]:

        driver.get(url=URL)

        add\_class=driver.find\_element\_by\_class\_name('add-classes')#class 추가

        for a in range(3):

            add\_class.click()

        for percent in range(0,101,25):

            file\_path=os.listdir('C:₩₩Users₩₩이채은₩₩OneDrive₩₩바탕 화면₩₩포트폴리오₩₩codingcoding₩₩project₩₩studydata\_'+key+"₩"+food+str(percent)) #폴더안의 파일 주소 가져오기

            upload\_=driver.find\_element\_by\_css\_selector("#sample-input-list > button:nth-child(2)")

            upload\_.click()

            studydata\_input=driver.find\_element\_by\_id("file-input")

            for study in file\_path:

                studydata\_input.send\_keys(study)

        start\_study=driver.find\_element\_by\_css\_selector("#container > tm-button")

        select = driver.find\_element\_by\_id('select-input')

        select.select\_by\_visible\_text("파일")

        sikpan\_upload=driver.find\_element\_by\_id('file-input') #결과도출&리스트 저장

        sikpan\_pic\_path=os.listdir(".₩₩Users₩₩이채은₩₩OneDrive₩₩바탕 화면₩₩포트폴리오codingcoding₩₩project₩₩studydata\_₩₩sikpan\_"+key+"₩₩"+food)

        for file in sikpan\_pic\_path:

            sikpan\_upload.send\_key(file)

            individual\_result=driver.find\_elements\_by\_id("value-label")

            menu\_result[key][menukey].append(individual\_result)

        menukey=menukey+1

        driver.close()

#%제거

for key in menu\_result:

    for n in key:

        for m in n:

            for number in m:

                m=float((m.replace("%","")))

print(menu\_result)

4. 그래프 그리기

우리는 사람들이 보통 메뉴를 몇 퍼센트 정도 먹는지 알기 위해 정규 분포 그래프를 그릴 것이다.

그래프를 그리는 데에는 numpy모듈과 matplotlib모듈을 사용했다.

먼저, 위 크롤링 과정을 거치면서 자료가 어떻게 저장 되어있는지를 알아야 그래프를 그릴 수 있기 때문에 우리의 자료구조를 알아보자.

위 코드에 이런 부분이 있다. 이건 딕셔너리라고 하는 자료구조이다. 그런데 여기에는 리스트가 들어있다. []이게 리스트다.

menu\_result={

    "day1":[[],[],[],[],[]],

    "day2":[[],[],[],[],[]]

    }

Day 1에 5개의 []이 있다. 이건 5개의 메뉴들이다. 그래서 여기에 사람들이 해당 메뉴를 얼마나 먹었는지 (%) 숫자가 들어갈 것이다.

그런데 여기 문제가 있다. 아래 사진을 보자. 결과창 사진 아래에 각각 class에 해당할 확률이 몇 퍼센트인지 나오게 된다. 우리는 class가 5개나 된다. 우리는 크롤링 과정에서 각각 class에 해당하는 확률값을 끌어왔기 때문에 한사람의 한메뉴 대한 결과 값이 [40,20,3,4,0] 이런식으로 저장이 된다. 아무튼 그래서 이거 하나의 값으로 바꿔야 그래프 그릴 수 있다는 말임 근데 우리 데이터 600개 정도 있는데 이거 노가다로 언제 다 처리함? 이때 바로 numpy모듈이 쓰이는 것임. Numpy를 이용해서 600개 넘는 데이터의 처리를 깔끔하게 해냈다.

{(class’0%먹음’에 해당하는 퍼센티지)\*0 +(class’25%먹음’에 해당하는 퍼센티지)\*25+(class’50%먹음’에 해당하는 퍼센티지)\*50+(class’75%먹음’에 해당하는 퍼센티지)\*75+(class’100%먹음’에 해당하는 퍼센티지)\*100} 을 100으로 나눈다면, 아마 그사람이 먹은 비율을 알 수 있을 것이다.

아래것은 위 설명한 부분에 해당하는 파이썬 코드이다. (이 또한 dataprocess.py에 있다)

for key in menu\_result:

    for n in key:

        np\_array=np.array(n)

        sum\_array=np\_array.sum(axis=1)

        target\_array=sum\_array/100

이제 데이터들이 정리가 된것같다. 메뉴별로, 사람들 각각이 얼마나 먹었는지. 이를 가지고 정규분포그래프를 그리기 위해 나는 파이썬의 matplotlib이라는 모듈을 사용할 것이다. plt.hist가 정규분포를 그려주는 명령어다.

for key in menu\_result:

    menukey=0

    for n in key:

        plt.title(menu[key][menukey]+"의 잔반")

        plt.hist(n)

        menukey=menukey+1

        plt.savefig('.₩₩'+menu[key][menukey]+'sin.png')

