2017160225 이도형 학기말과제

(1)

#Assignment 1

!pip install selenium

!apt install chromium-chromedriver

import time

from selenium import webdriver

import requests

from bs4 import BeautifulSoup

number = input()

url = 'https://m.dhlottery.co.kr/gameResult.do?method=allWin'

header\_info = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36 Edge/18.19582'}

page = requests.get(url, headers = header\_info)

page = page.text

specific\_options = webdriver.ChromeOptions()

specific\_options.add\_argument('--headless')

specific\_options.add\_argument('--no-sandbox')

driver = webdriver.Chrome('chromedriver', options = specific\_options)

driver.get(url)

time.sleep(1)

start\_box = driver.find\_element\_by\_name('drwNoStart') # name으로 start\_box 찾기

end\_box = driver.find\_element\_by\_name('drwNoEnd') # name으로 end\_box 찾기

start\_box.send\_keys(str(number))

end\_box.send\_keys(str(number))

search\_xpath = '//\*[@id="searchBtn"]/span'

search = driver.find\_element\_by\_xpath(search\_xpath)

search.click()

time.sleep(1)

soup = BeautifulSoup(driver.page\_source, 'html.parser')

people = soup.find\_all('strong')

# soup을 반드시 xpath를 이용해 search를 한 후 만들고, find\_all 해주어야 한다.

# 이유 : url이 안 바뀌더라도 search에 따라 html이 바뀔 수 있는데, 이 변화를 반영해서 soup해야하기 때문.

stronglist=[]

for element in people:

  txel = element.text

  txel = txel.strip()

  stronglist.append(txel)

pplnum = stronglist[1].split()

print(pplnum[1])

# 사람 수는 strong tag 중 두번째에 존재함.

# 따라서, 텍스트를 잘라서 만든 리스트의 [1]요소를 취해주어 사람 수를 알 수 있다.

# Assignment 2

!pip install selenium

!apt install chromium-chromedriver

import time

from selenium import webdriver

import requests

from bs4 import BeautifulSoup

year=input()

month=input()

day=input()

if int(month)<10:

  month = '0'+month

if int(day)<10:

  day= '0'+day

# 해당 사이트에는 9월 이하, 9일 이하는(예: 2022/5/5) 5월 5일이 아니라 05월 05일로 입력해야함.

# 하지만 입력 예시는 2022 5 5의 같은 입력은 5를 05의 string으로 바꿔주는 작업이 필요.

url = 'https://www.nielsenkorea.co.kr/tv\_terrestrial\_day.asp?menu=Tit\_1&sub\_menu=1\_1&area=01'

header\_info = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36 Edge/18.19582'}

page = requests.get(url, headers = header\_info)

page = page.text

specific\_options = webdriver.ChromeOptions()

specific\_options.add\_argument('--headless')

specific\_options.add\_argument('--no-sandbox')

driver = webdriver.Chrome('chromedriver', options = specific\_options)

driver.get(url)

time.sleep(1)

year\_box = driver.find\_element\_by\_name('sYear')

month\_box = driver.find\_element\_by\_name('sMonth')

day\_box = driver.find\_element\_by\_name('sDay')

year\_box.send\_keys(year)

month\_box.send\_keys(month)

day\_box.send\_keys(day)

search\_xpath = '//\*[@id="sub\_body"]/table[1]/tbody/tr/td/form/table/tbody/tr/td[2]/img'

search = driver.find\_element\_by\_xpath(search\_xpath)

search.click()

time.sleep(1)

soup = BeautifulSoup(driver.page\_source, 'html.parser')

programs = soup.find\_all(class\_='tb\_txt')

# 'tb\_txt' class를 find\_all 해주면 방영 목록이 크롤링됨.

programlist = []

for program in programs:

  protxt = program.text

  protxt = protxt.strip()

  programlist.append(protxt)

# 이 방영 목록을 txt로 자르고 strip하여 programlist에 append시킨 후

print(programlist[0])

print(programlist[1])

print(programlist[2])

# 처음 세 개를 print해주면 됨.

# Assignment 3

import pandas as pd

df = pd.read\_csv('score.csv')

highstd= int(df.shape[0]\*0.75)-1

lowstd= int(df.shape[0]\*0.25)+1

# 상위 25프로, 하위 25프로에 해당되는 학생의 점수는 quantile과는 다르다.

# 따라서, n%에 해당되는 학생의 점수는 위와 같은 방법으로 index를 추출 후

scorelist=sorted(list(df['TotalScore']))

high=scorelist[highstd]

low=scorelist[lowstd]

isangchi = (high - low)\*1.5

# scorelist에서 추출한 index의 score로 점수를 파악

cond1 = df['TotalScore'] >= high+isangchi

cond2 = df['TotalScore'] <= low-isangchi

cond = cond1 | cond2

df\_new = df[cond]

# 두 조건의 or 조건으로 새로운 datafram을 만들고 (이 때 성적은 매우 높거나 매우 낮거나)

for i in range(0, df\_new.shape[0]):

  if df\_new.iloc[i, 4]<=low-isangchi: # 낮은 경우에는 '할 수 있어요', 그 외에는 '대단합니다'출력

    print("{} 학생 할 수 있어요!".format(df\_new.iloc[i,0]))

  else:

    print("{} 학생 대단합니다!".format(df\_new.iloc[i,0]))

# Assignment 4

!pip install konlpy

from konlpy.tag import Komoran

text = input()

filter = ['~', '!', '@', '\*', '%', '^', '&', '?']

for i in filter:

  text=text.replace(i, '')

# 특수문자를 제거하는 전처리 적용

komoran = Komoran()

result = komoran.morphs(text)

pos\_result = komoran.pos(text)

# text를 형태소 단위로 분리, tagging. 이 때 pos\_result list의 각요소는 (word, 품사)의 tupule

poslist=[]

for i in pos\_result:

  word, pos = i # pos\_result의 각 요소 tupule을 word, pos로 저장.

  if pos in ['NNG', 'NNP', 'NNB', 'NP', 'NR', 'MAG', 'MAJ', 'SN']: #pos가 본 리스트(명사, 대명사, 부사, 숫자) 안에 있을 경우

    poslist.append(word) #word를 새로운 리스트에 넣어줌

print(sorted(poslist)) # 오름차순으로 정리된 poslist를 출력.

# Assignment 5

import pandas as pd

import matplotlib.pyplot as plt

# (1)

df = pd.read\_excel('job.xlsx')

cond1=df['자치구']=='노원구'

df\_1 = df[cond1]

# 노원구에 해당되는 dataframe만 추출

df\_1.plot('기간', '취업률') # 이 새로운 dataframe의 기간과 취업률을 plotting

# (2)

cond21 = df['기간']==2014

cond22 = df['자치구'] == '용산구'

cond23 = df['자치구'] == '서초구'

cond24 = df['자치구'] == '강남구'

cond2 = cond21 & (cond22 | cond23 | cond24)

# 2013년의 용산구+서초구+강남구의 condition cond2로 지정

df\_2 = df[cond2]

# 해당 cond2대로 filtering하여 df\_2로 지정하고

df\_2.plot('자치구', '취업자 수', kind='bar')

# bar 형태로 자치구 별 취업자 수를 fitting

# (3)

cond3 = df['기간']==2017

df\_3 = df[cond3]

# 2017년의 dataframe을 df\_3에 저장하고

df\_3.plot('졸업자 수', '취업자 수', kind='scatter')

# df\_3의 졸업자 수와 취업자 수를 scatter형으로 fitting.

(2)는 소스코드에 주석으로 표시하였습니다.

(3)

텍스트이(가) 표시된 사진

자동 생성된 설명

그림 Question 1 Result

텍스트이(가) 표시된 사진

자동 생성된 설명

그림 Question 2 Result

텍스트이(가) 표시된 사진

자동 생성된 설명

그림 Question 3 Result

텍스트이(가) 표시된 사진

자동 생성된 설명

그림 Question 4 Result

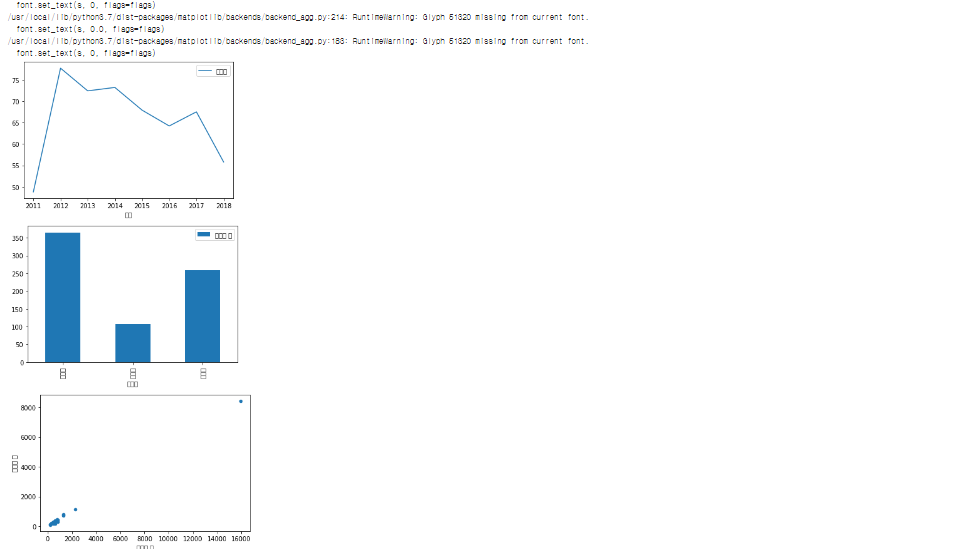


그림 Question 5 Result