Portfolio

●●●

Jinyoung Kim

https://github.com/kim715go/portfolio

Contents

Real-World Data Handling Experience

Cleansing/Interpolation of Electricity Usage Data

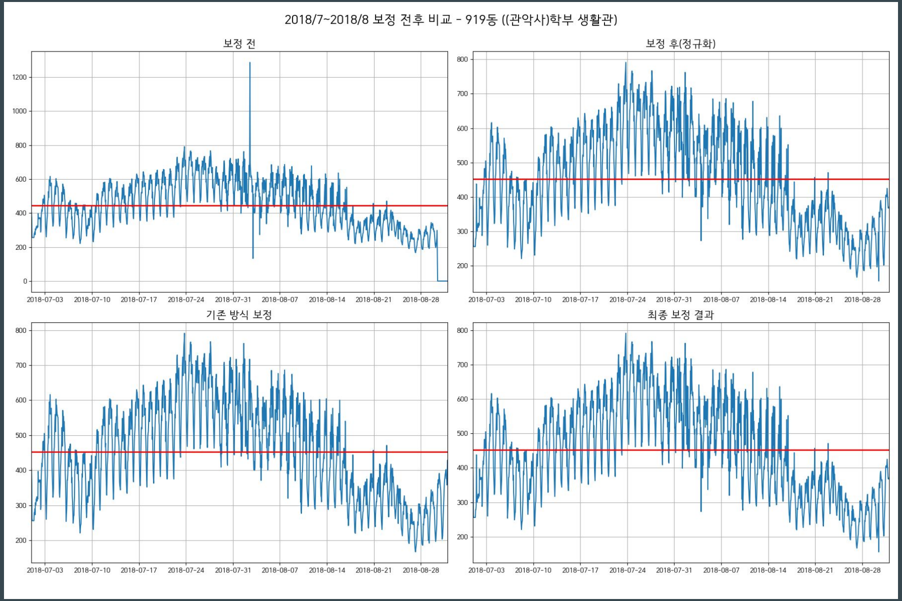
- Given data
  - Hourly electricity usage data per building (total 156 buildings)
- Types of errors
  - Unrealistically large or negative usage
  - Complete loss of data
  - Repeated usage for too many times
- Reason of errors
  - Malfunction of individual meters
  - Connection loss between meters and the control server
  - Other unknown reasons
- Rate of errors
  - Some 10% with minor fluctuation month by month
  - After the cleansing process, more than 90% of the errors are replaced with plausible values

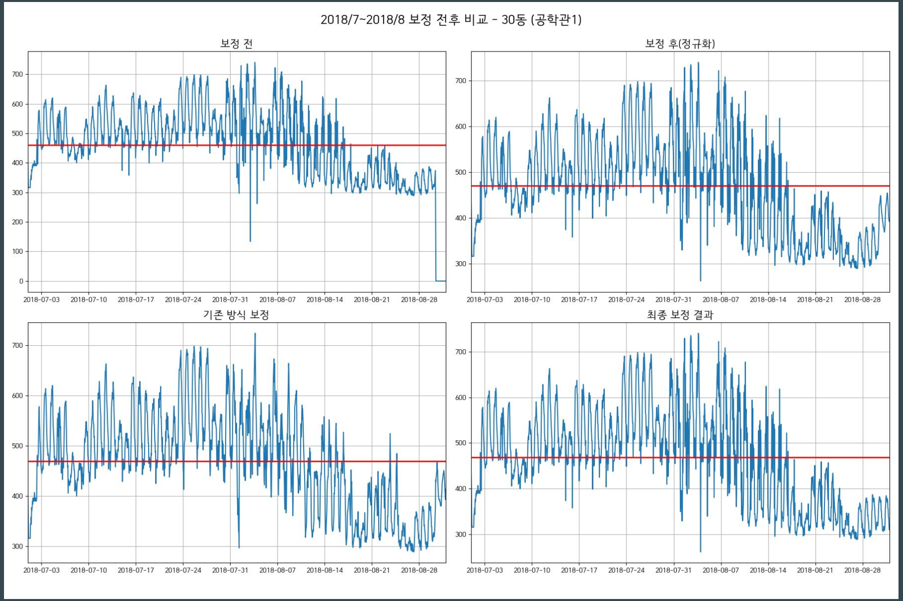
Cleansing/Interpolation of Electricity Usage Data

- Error detection strategies
  - (variance of difference) Using the fact that electricity usage tends to change continuously, consider a usage value with a large gap between that of the previous hour as an error.
  - (confidence interval) For every value, sample the values ±1 hour, ±1 day, and ±1~4 years to make a sample group and get the 98% confidence interval from the group. If the value is out of the range, consider it an error.
- Interpolation (correction) strategies
  - (weighted mean) For every erroneous value, sample the values ±1~4 weeks and ±1~2 years to make a sample group, and get a weighted mean as a replacement. weights are given inversely proportional to time distance from the erroneous value.
  - (mean) Use the mean value or the median of a sample group as a replacement.

Cleansing/Interpolation of Electricity Usage Data

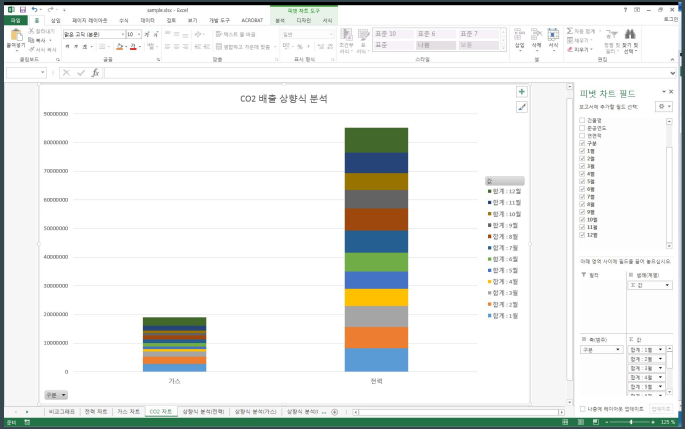
- Sample comparison graphs (drawn by python matplotlib)

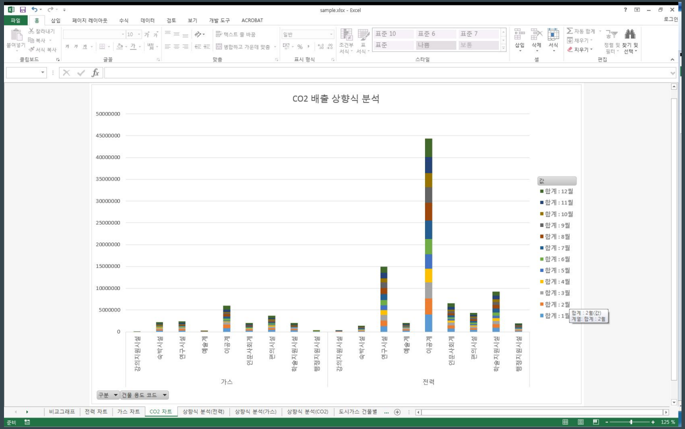




EDA/Statistical Approach to Energy Usage Data

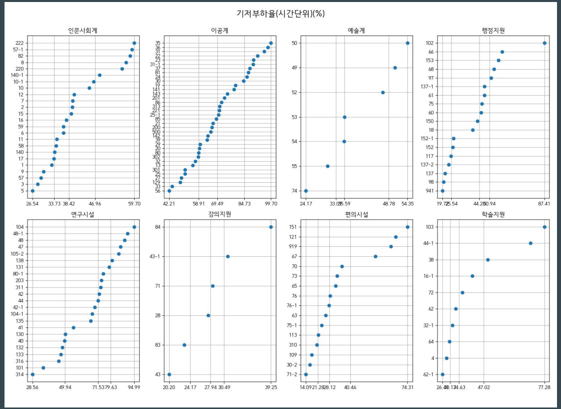
- CO2 emission status visualization
    - Using hourly electricity usage data per building and hourly natural gas usage data per building, calculate the CO2 emission value.
    - Summarizing the value according to the metadata of the buildings, such as department, purpose, gross floor area, etc.
    - Sample graphs in Excel

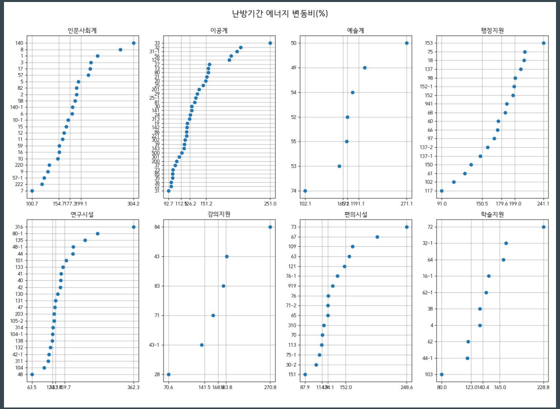


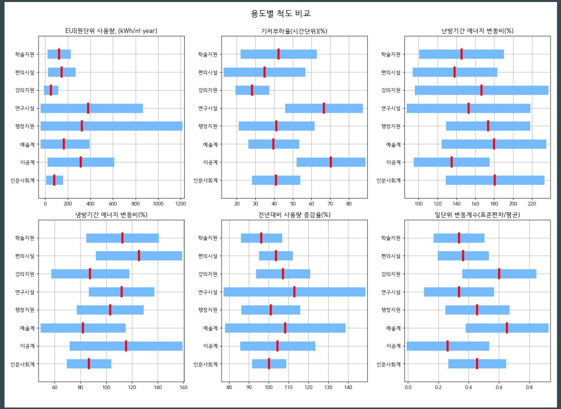


EDA/Statistical Approach to Energy Usage Data

- Characteristic index analysis for each building
  - With the help of metadata of the buildings and utilizing some data-driven methods such as EDA(exploratory data analysis) and PCA(principal component analysis), several indices that represent the characteristics of electricity use of a building were designed.
  - Samples of the result graphs

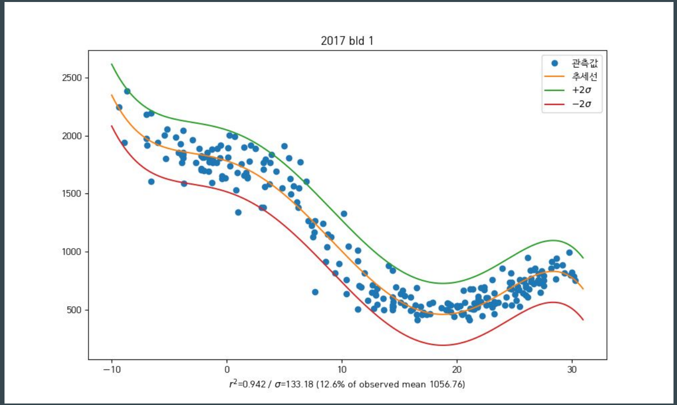


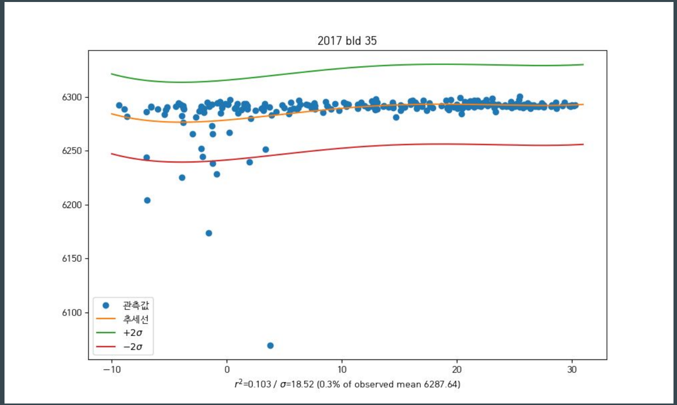




EDA/Statistical Approach to Energy Usage Data

- Correlation/regression analysis
  - Goal : to understand the characteristics of patterns of energy usage(especially electricity)
  - Domain Knowledge : electricity usage is always high on summers and winters - temperature can be a major factor deciding the amount of energy use
  - Variables : daily mean temperature and daily electricity usage value per building
  - Regression formula : 5th polynomial (useful to detect inflection points)
  - Sample graphs



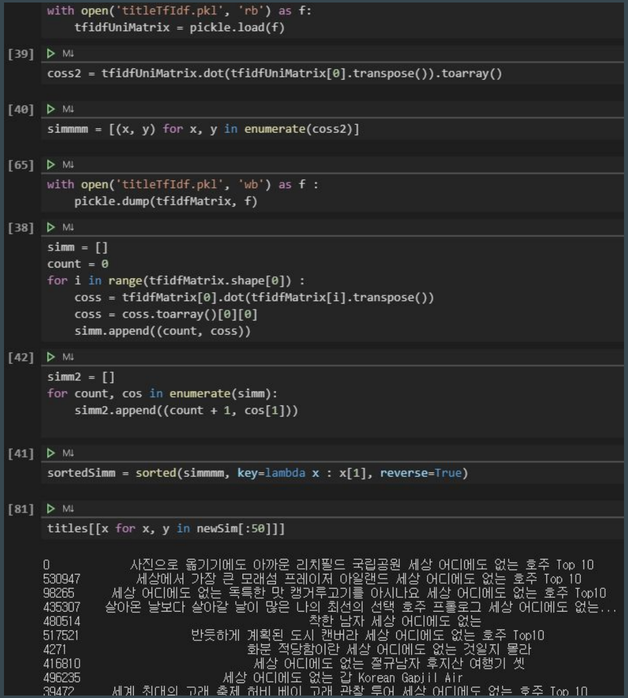


Machine Learning with Kakao Brunch Article Data

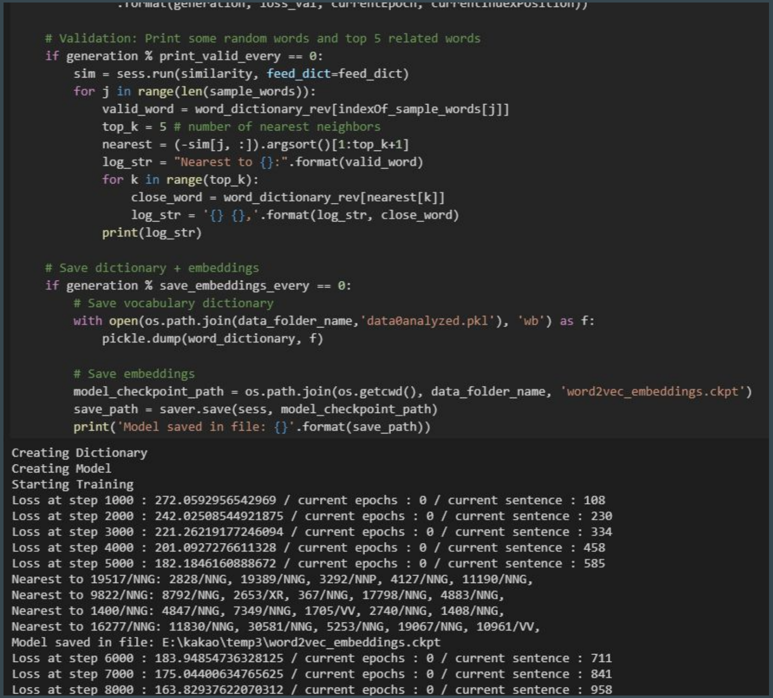
- Goals
  - To design a measure to check the similarity among articles to make a recommendation to Brunch users
- Given data
  - A list of Brunch users and their followers
  - A list of Brunch articles and tokenized text of them
  - Access record of articles in time sequence
- Procedures
  - To make a tf-idf matrix using the titles and the snippets of the given articles
  - To use Tensorflow library, implement a doc2Vec machine learning algorithm and obtain a document matrix
  - To utilize cosine similarity when checking the (virtual) distance between documents
  - To make use of the result for recommending relevant articles customized to each user

Machine Learning with Kakao Brunch Article Data

- Similarity analysis with tf-idf matrices(left) / Doc2vec modeling(right)



```
    with open('titleTfIdf.pkl', 'rb') as f:
        tfidfUniMatrix = pickle.load(f)

[39]
    coss2 = tfidfUniMatrix.dot(tfidfUniMatrix[0].transpose()).toarray()

[40]
    simmmm = [(x, y) for x, y in enumerate(coss2)]

[65]
    with open('titleTfIdf.pkl', 'wb') as f :
        pickle.dump(tfidfMatrix, f)

[38]
    simm = []
    count = 0
    for i in range(tfidfMatrix.shape[0]) :
        coss = tfidfMatrix[0].dot(tfidfMatrix[i].transpose())
        coss = coss.toarray()[0][0]
        simm.append((count, coss))

[42]
    simm2 = []
    for count, cos in enumerate(simm):
        simm2.append((count + 1, cos[1]))

[41]
    sortedSimm = sorted(simmmm, key=lambda x : x[1], reverse=True)

[81]
    titles[[x for x, y in newSim[:50]]]

    0              사진으로 옮기기에도 아까운 리치필드 국립공원 세상 어디에도 없는 호주 Top 10
    530947          세상에서 가장 큰 모래섬 프레이저 아일랜드 세상 어디에도 없는 호주 Top 10
    98265       세상 어디에도 없는 독특한 맛 캥거루고기를 아시나요 세상 어디에도 없는 호주 Top10
    435307     살아온 날보다 살아갈 날이 많은 나의 최선의 선택 호주 프롤로그 세상 어디에도 없는...
    480514                                  착한 남자 세상 어디에도 없는
    517521                  반듯하게 계획된 도시 캔버라 세상 어디에도 없는 호주 Top10
    4271                              화분 적당함이란 세상 어디에도 없는 것일지 몰라
    416810                             세상 어디에도 없는 절규남자 후지산 여행기 셋
    496235                        세상 어디에도 없는 갑 Korean Gapjil Air
    39472       세계 최대의 고래 축제 허비 베이 고래 관찰 투어 세상 어디에도 없는 호주 Top 10
```



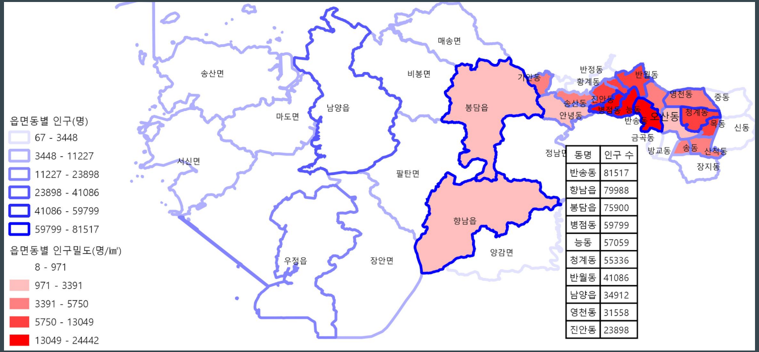
```
            .format(generation, loss_val, currentEpoch, currentIndexPosition))

    # Validation: Print some random words and top 5 related words
    if generation % print_valid_every == 0:
        sim = sess.run(similarity, feed_dict=feed_dict)
        for j in range(len(sample_words)):
            valid_word = word_dictionary_rev[indexOf_sample_words[j]]
            top_k = 5 # number of nearest neighbors
            nearest = (-sim[j, :]).argsort()[1:top_k+1]
            log_str = "Nearest to {}:".format(valid_word)
            for k in range(top_k):
                close_word = word_dictionary_rev[nearest[k]]
                log_str = '{} {},'.format(log_str, close_word)
            print(log_str)

    # Save dictionary + embeddings
    if generation % save_embeddings_every == 0:
        # Save vocabulary dictionary
        with open(os.path.join(data_folder_name,'data0analyzed.pkl'), 'wb') as f:
            pickle.dump(word_dictionary, f)

        # Save embeddings
        model_checkpoint_path = os.path.join(os.getcwd(), data_folder_name, 'word2vec_embeddings.ckpt')
        save_path = saver.save(sess, model_checkpoint_path)
        print('Model saved in file: {}'.format(save_path))
Creating Dictionary
Creating Model
Starting Training
Loss at step 1000 : 272.0592956542969 / current epochs : 0 / current sentence : 108
Loss at step 2000 : 242.02508544921875 / current epochs : 0 / current sentence : 230
Loss at step 3000 : 221.26219177246094 / current epochs : 0 / current sentence : 334
Loss at step 4000 : 201.0927276611328 / current epochs : 0 / current sentence : 458
Loss at step 5000 : 182.1846160888672 / current epochs : 0 / current sentence : 585
Nearest to 19517/NNG: 2828/NNG, 19389/NNG, 3292/NNP, 4127/NNG, 11190/NNG,
Nearest to 9822/NNG: 8792/NNG, 2653/XR, 367/NNG, 17798/NNG, 4883/NNG,
Nearest to 1400/NNG: 4847/NNG, 7349/NNG, 1705/VV, 2740/NNG, 1408/NNG,
Nearest to 16277/NNG: 11830/NNG, 30581/NNG, 5253/NNG, 19067/NNG, 10961/VV,
Model saved in file: E:\kakao\temp3\word2vec_embeddings.ckpt
Loss at step 6000 : 183.94854736328125 / current epochs : 0 / current sentence : 711
Loss at step 7000 : 175.04400634765625 / current epochs : 0 / current sentence : 841
Loss at step 8000 : 163.82937622070312 / current epochs : 0 / current sentence : 958
```

Analysis on Transport Card Data and Relevant Bus Routes

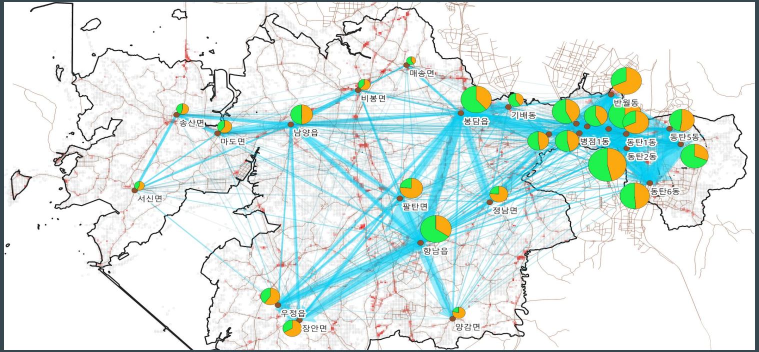
- Goals
  - To examine the status of bus service in Hwaseong, a city in Gyeonggi Province
  - To present realistic methods to improve the current system
- Given data
  - Geojson file of the road network in Hwaseong and its vicinity
  - Geojson file of the distribution of floating population and residents in Hwaseong
  - Transport card use data of 1st ~ 4th of July, 2018
  - Information on the current bus routes operating in Hwaseong and Suwon (incomplete)
- Procedures
  - Parse the given geojson files to make them easier to manipulate in QGIS
  - Obtain position data of the bus routes from Naver Map
  - Pre-process the transport card use data (removing rows with missing values, dropping columns unnecessary for analysis, etc)
  - Overlap different kinds of the shape files in QGIS to get an insight

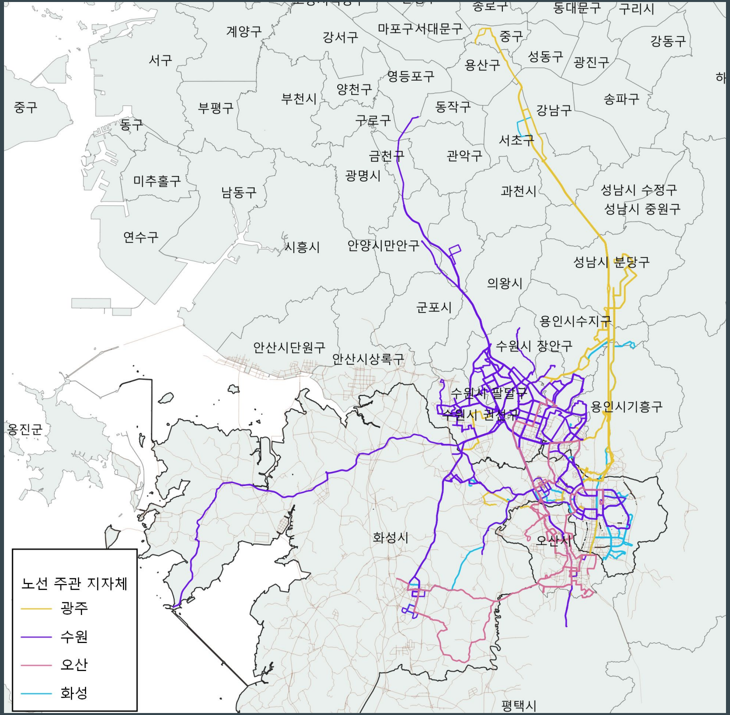
Analysis on Transport Card Data and Relevant Bus Routes

- Population density and flow analysis(left) / Bus route analysis(right)



동명	인구 수
반송동	81517
향남읍	79988
봉담읍	75900
병점동	59799
능동	57059
청계동	55336
반월동	41086
남양읍	34912
영천동	31558
진안동	23898





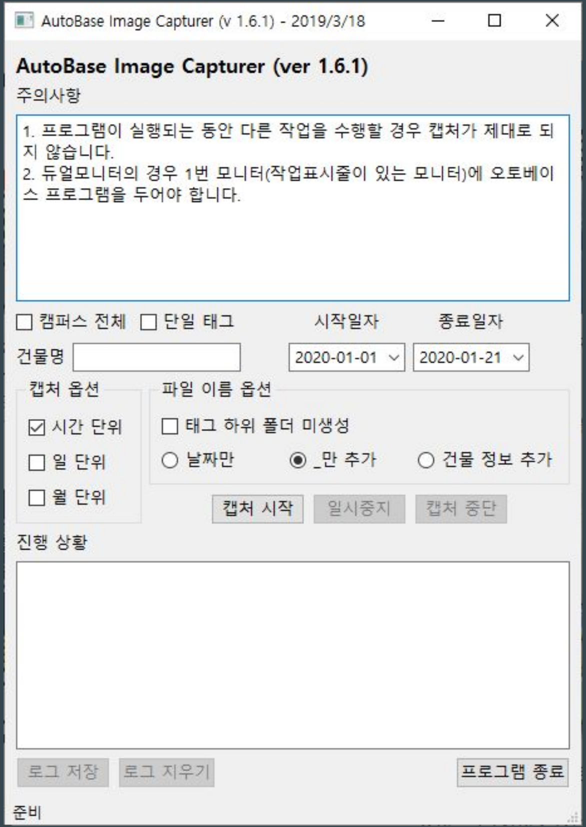
Development of Customized Programs

Automatic AutoBase Capturer

- What is AutoBase?
    - An operating program of electricity meters (in this case, meters installed in the school campus)
    - An end user can see information on each meter and check the usage, but cannot download it
- Goals
    - To save screens of the program showing electricity usage without human intervention
- Conditions
    - AutoBase only works under the intranet of the campus
    - Buildings with meters are shown on the campus map, and an end user can click the building to see the meters. The meters should be clicked to see the usage data.
    - Other members of the office do not have any knowledge on programming
- Strategies
    - To use PyQt5 library to make a GUI program with Python
    - For ease of development, the program gets the whole control of a mouse and a keyboard of a computer when it is working

Automatic AutoBase Capturer

- Program interface
  - Video of actual use → https://youtu.be/zM8Vzt6fT-8



AutoBase Image Capturer (v 1.6.1) - 2019/3/18

AutoBase Image Capturer (ver 1.6.1)

주의사항

1. 프로그램이 실행되는 동안 다른 작업을 수행할 경우 캡처가 제대로 되지 않습니다.
2. 듀얼모니터의 경우 1번 모니터(작업표시줄이 있는 모니터)에 오토베이스 프로그램을 두어야 합니다.

☐ 캠퍼스 전체 ☐ 단일 태그 시작일자 2020-01-01 종료일자 2020-01-21

건물명

캡처 옵션: ☑ 시간 단위 ☐ 일 단위 ☐ 월 단위

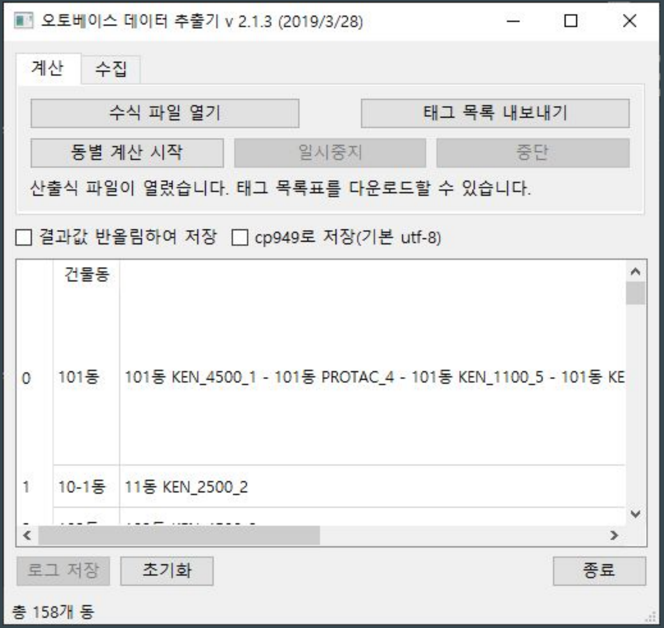
파일 이름 옵션: ☐ 태그 하위 폴더 미생성 ○ 날짜만 [✓ _만 추가] ○ 건물 정보 추가

Electricity Usage Data Crawler and Calculator

- Issues
  - AutoBase does show the usage values, but it does not have any function to download the data.
  - It is often needed to calculate the values from different meters to get the usage of a single building
- Goals
  - Automatically crawl the electricity usage data on-demand
  - Calculate the meter-wise data to obtain building-wise usage data according to the given formulae
- Conditions
  - AutoBase gets the usage data through http from the server.
- Strategies
  - With a web debugger (Fiddler in this case), find the packet that AutoBase sends to request the required data and mimic the pattern with Python Requests library
  - Use a single format for saving the usage values on meters and buildings alike
  - To use PyQt5 library to make a GUI program with Python

Electricity Usage Data Crawler and Calculator

- Program interface
  - Video of actual use → https://youtu.be/AjSUMGZaC8A

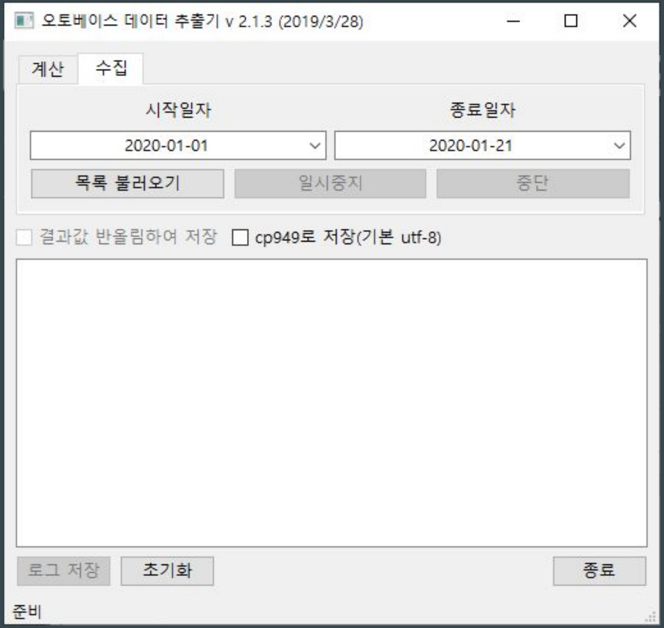


산출식 파일이 열렸습니다. 태그 목록표를 다운로드할 수 있습니다.

☐ 결과값 반올림하여 저장 ☐ cp949로 저장(기본 utf-8)

	건물동	
0	101동	101동 KEN_4500_1 - 101동 PROTAC_4 - 101동 KEN_1100_5 - 101동 KE
1	10-1동	11동 KEN_2500_2

총 158개 동



시작일자	종료일자
2020-01-01	2020-01-21

☐ 결과값 반올림하여 저장 ☐ cp949로 저장(기본 utf-8)

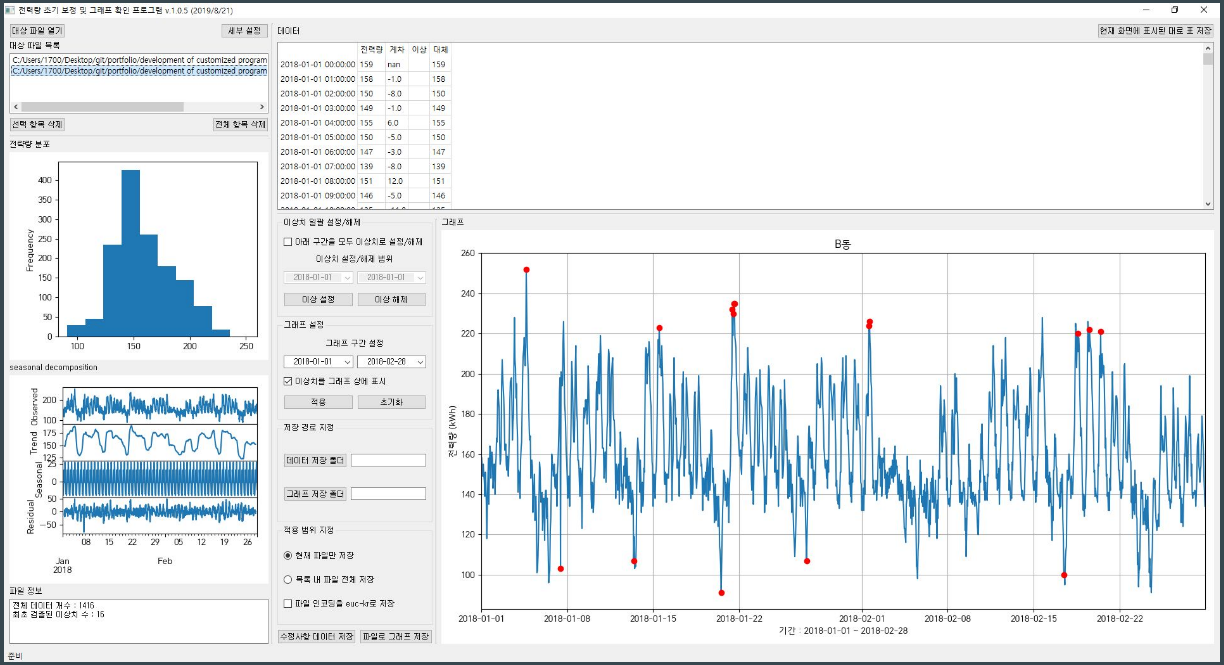
준비

Electricity Data Checker

- Issues
  - Plotting graphs of hourly electricity usage data gets trickier as the data cumulates.
  - Though data cleansing process is done automatically, some of errors are still undetected.
- Goals
  - To provide a fast and convenient tool for plotting graphs of the usage data and replace undetected but invalid values with a zero, which is always detected as an error during cleansing process.
- Strategies
  - To use PyQt5 library to make a GUI program with Python
  - To implement seasonal decomposition, a method used for time series analysis, to aid the examination of possible errors
  - To allow a user to choose which encoding to use when saving the result as a csv file, considering that Excel is the main program in the center. (csv and text files saved with Unicode are tricky to open with Excel)

Electricity Data Checker

- Program interface
    - Video of actual use → https://youtu.be/s6ttk3NJbQs



C:/Users/1700/Desktop/git/portfolio/development of customized program
C:/Users/1700/Desktop/git/portfolio/development of customized program

	전력량	계차	이상	대체
2018-01-01 00:00:00	159	nan		159
2018-01-01 01:00:00	158	-1.0		158
2018-01-01 02:00:00	150	-8.0		150
2018-01-01 03:00:00	149	-1.0		149
2018-01-01 04:00:00	155	6.0		155
2018-01-01 05:00:00	150	-5.0		150
2018-01-01 06:00:00	147	-3.0		147
2018-01-01 07:00:00	139	-8.0		139
2018-01-01 08:00:00	151	12.0		151
2018-01-01 09:00:00	146	-5.0		146

[ ] 아래 구간을 모두 이상치로 설정/해제

이상치 설정/해제 범위: 2018-01-01 ~ 2018-01-01

그래프 구간 설정: 2018-01-01 ~ 2018-02-28

[✓] 이상치를 그래프 상에 표시

(●) 현재 파일만 저장

( ) 목록 내 파일 전체 저장

[ ] 파일 인코딩을 euc-kr로 저장

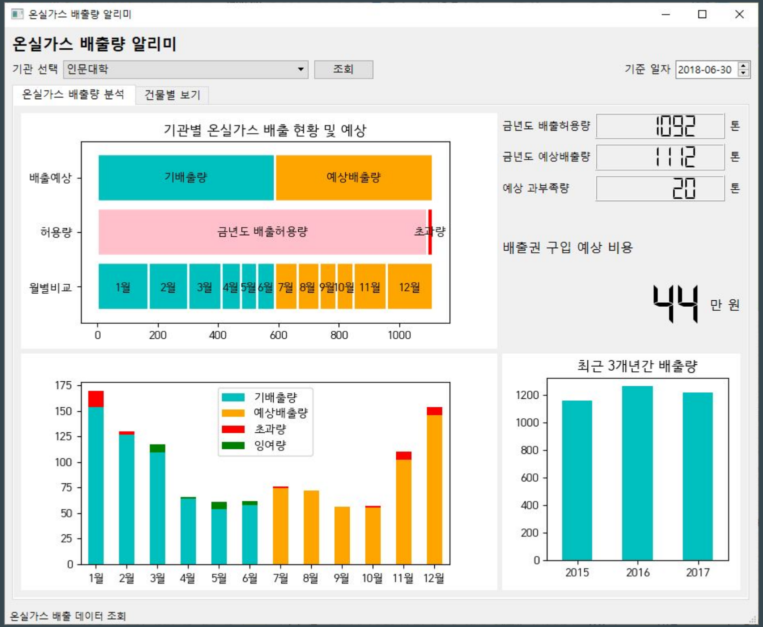
전체 데이터 개수 : 1416
최초 검출된 이상치 수 : 16

CO2 Emission Checker

- Issues
  - Due to the advent of emission trading, the necessity of notifying the amount of CO2 emissions to the administrators of the departments in the campus.
- Goals
  - To make an easy-to-understand program showing the CO2 emission per department
- Conditions
  - CO2 emission by electricity and natural gas must be combined properly
  - CO2 emission should be shown per department and per building when necessary
  - Distributable version of the program is to be developed by an outside company
- Strategies
  - To make a prototype of the program with Python to review its composition
  - To provide the prototype to the outside software company for reference

CO2 Emission Checker

- Program interface
  - Video of actual use → https://youtu.be/F09TqxCIGA4



온실가스 배출량 알리미

온실가스 배출량 알리미

기관 선택 인문대학 조회 기준 일자 2018-06-30

온실가스 배출량 분석 | 건물별 보기

금년도 배출허용량 1092 톤

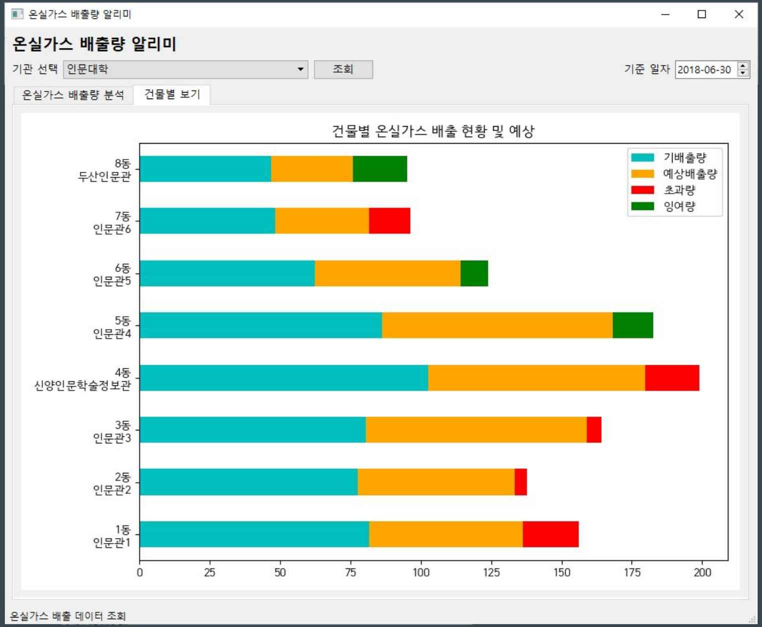
금년도 예상배출량 1112 톤

예상 과부족량 20 톤

배출권 구입 예상 비용

44 만 원

온실가스 배출 데이터 조회



온실가스 배출량 알리미

온실가스 배출량 알리미

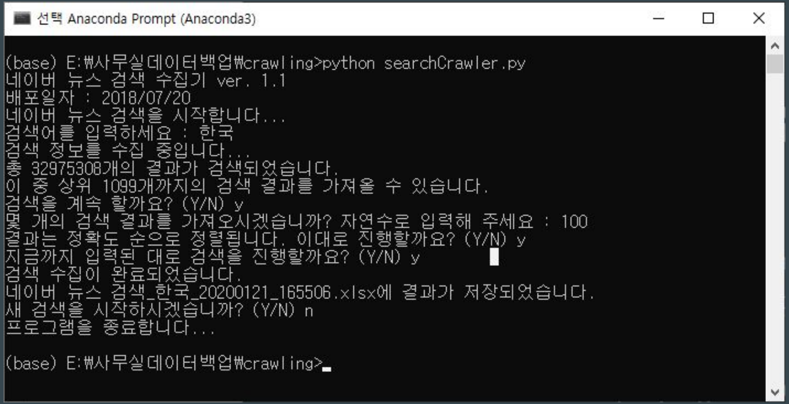
기관 선택 인문대학 조회 기준 일자 2018-06-30

온실가스 배출량 분석 | 건물별 보기

온실가스 배출 데이터 조회

Naver News Article Crawler

- Issues
  - While participating in a research group, the need to construct database on conflicts among residents on property development was raised.
- Goals
  - To gather news articles of a topic on Naver News
- Strategies
  - To utilize an open API of news search provided for developers by Naver
  - To make a CUI program to shorten the development period
  - To package the Python code file into an exe file with Pyinstaller library
- Paper
  - A study on conflict DB construction using python-based web-crawling (2018, participated as a co-author)



```
(base) E:₩사무실데이터백업₩crawling>python searchCrawler.py
네이버 뉴스 검색 수집기 ver. 1.1
배포일자 : 2018/07/20
네이버 뉴스 검색을 시작합니다...
검색어를 입력하세요 : 한국
검색 정보를 수집 중입니다...
총 32975308개의 결과가 검색되었습니다.
이 중 상위 1099개까지의 검색 결과를 가져올 수 있습니다.
검색을 계속 할까요? (Y/N) y
몇 개의 검색 결과를 가져오시겠습니까? 자연수로 입력해 주세요 : 100
결과는 정확도 순으로 정렬됩니다. 이대로 진행할까요? (Y/N) y
지금까지 입력된 대로 검색을 진행할까요? (Y/N) y
검색 수집이 완료되었습니다.
네이버 뉴스 검색_한국_20200121_165506.xlsx에 결과가 저장되었습니다.
새 검색을 시작하시겠습니까? (Y/N) n
프로그램을 종료합니다...

(base) E:₩사무실데이터백업₩crawling>
```

Contest

Smart City Simulation Contest

- Overview of the contest
  - Held in 2018 by LH(Korea Land and Housing Corporation)
  - Participants were expected to freely design 5-1 district of Sejong City under the basic concept and development plan of the place and represent it with Cities: Skylines, a city-builder simulation game.
  - The grand prize winner was expected to revise their work according to requests from the host and submit a 5-minute complete video clip of their work on Cities: Skylines
- Participation progress and result
  - Participated as a team of two (major role of mine was to construct the city on the game and edit the video clip)
  - Title of work was “Baraon(바라온)” and its major concept was “An Innovative Space Planning Resembling Artificial Neural Network Structure”
  - Awarded the grand prize with 10,000,000 KRW as a reward
  - Conducted the following work until late 2019