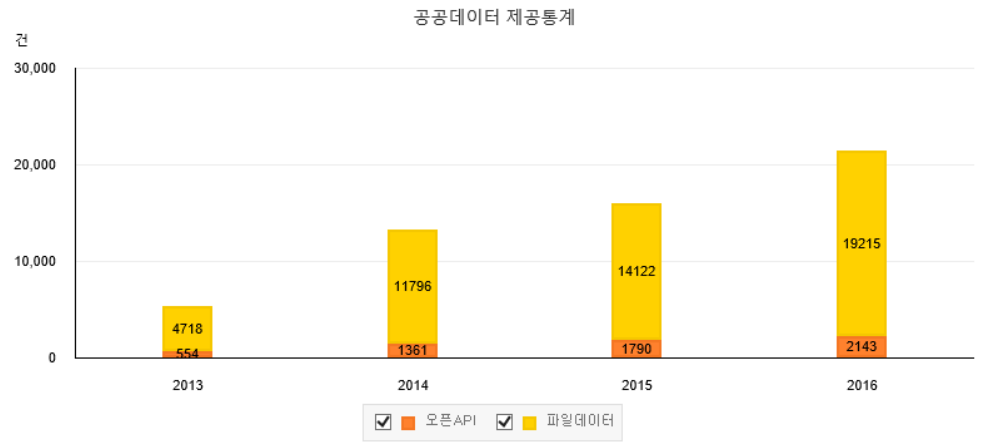
공공데이터포털 분석
- 데이터 수집 및 전처리

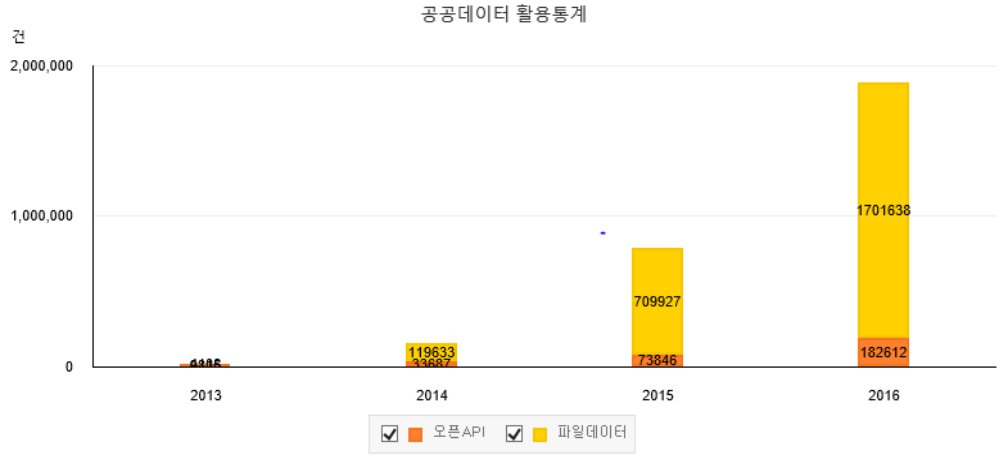
강원대학교 글로벌비즈니스학과 김도훈
한국과학기술정보연구원 김학래

Contents

- 분석의 필요성
- 데이터 전처리
- 데이터 분석
- 추후 계획

1) 공공 데이터 포탈 현황





[단위 : 건(누적)]

	2013		2014		2015		2016	
	공공데이터개방통계	공공데이터활용통계	공공데이터개방통계	공공데이터활용통계	공공데이터개방통계	공공데이터활용통계	공공데이터개방통계	공공데이터활용통계
소계	5,272	13,923	13,157	153,320	15,912	783,773	21,358	1,884,250
오픈API	554	9,815	1,361	33,687	1,790	73,846	2,143	182,612
파일데이터	4,718	4,108	11,796	119,633	14,122	709,927	19,215	1,701,638

포탈에 업로드 되는 데이터의 양은 증가하고 있음

2) 공공 데이터 문제점

(a) 통일 되지 않은 양식 / 단위

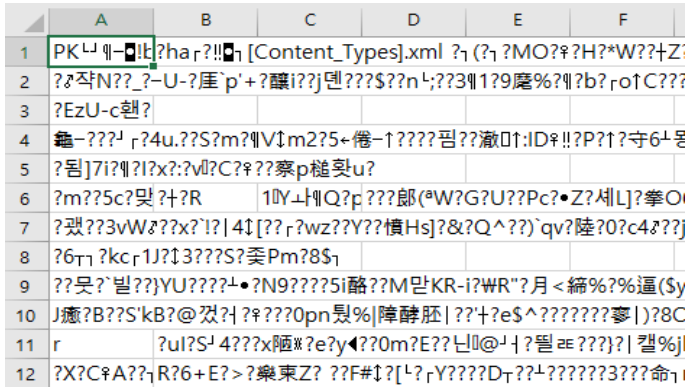
같은 단어 / 같은 의미의 단어도 다르게 표현

날짜	요일	연도	Date

연락처	문의연락처	전화	전화번호

(b) 파일 오류

글자 깨짐, 값이 없는 파일 등



(c) 파일 업데이트

비주기적인 파일 업데이트

경상북도 영양군_어린이보호구역_20170814			
업데이트 주기	수시	차기등록예정일	2018-07-01
경상북도 영양군_어린이보호구역_20150914			
업데이트 주기	연간	차기등록예정일	2017-08-01

(d) 데이터 요청

요청한 데이터를 얻기 어려움

신청 공공데이터 명칭	신청일자	기관담당자	접수기관	처리기한	처리상태
위경도, 사고차량, 사고자 정보등이 포함된 스쿨존 내 어린이 교통사고 데이터 / 스쿨존 내 과속방지 카메라와 방지턱 데이터	2017-07-11	박해수	도로교통공단	2017-07-24	제공
위경도, 사고차량, 사고자 정보등이 포함된 스쿨존 내 어린이 교통사고 데이터 / 스쿨존 내 과속방지 카메라와 방지턱 데이터	2017-07-11	ciamsunset	경찰청	2017-07-24	제공신청취소
2013-2017 대학수학능력평가 및 6,9월 모의고사 언어영역 문항별 정답률 및 출제범위	2017-04-24		교육부	2017-05-10	제공신청 반려
대학수능능력시험 및 6월 9월 모의고사 등급별 원점수 커트라인	2017-04-24		교육부	2017-05-10	제공신청 반려

1) CSV 파일 데이터 전처리

	A	B	C
1	목록명	데이터명	다운로드 URL
2	국가산업단지 산업동향정보	2012년 5월 국가산업단지 산업동향	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000001212856&fileDetailSn=1&publicDataDetailPk=uddi:07b44140-4ded-40e6-946e-c03b317b833e
3	인구현황	2013년 9월 인구현황	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000001215215&fileDetailSn=1&publicDataDetailPk=uddi:07c50b21-6f66-4943-b405-8fdf4adec661
4	화성시 상수도공무대행업체 현황	상수도 공무대행업체 현황	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000001215878&fileDetailSn=1&publicDataDetailPk=uddi:07c9308d-3382-48ca-bafe-ad3990342e77
5	학생범죄자 부모관계	학생범죄자 부모관계(2009)	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000000770335&fileDetailSn=0&publicDataDetailPk=uddi:07cf33ac-4f9c-448f-b6c7-c94be59f544e
6	교통사고통계	시도별 월별 교통사고	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000001211830&fileDetailSn=1&publicDataDetailPk=uddi:07ea87d2-e41c-4afe-95df-b6d629d8b0e3
7	범죄자 처분결과	범죄자 처분결과(계)(2010)	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000000770339&fileDetailSn=0&publicDataDetailPk=uddi:0806a928-8b4b-47e7-8f25-f3462768dd53
8	하수도 시설현황	하수도 시설현황	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000001215501&fileDetailSn=1&publicDataDetailPk=uddi:08200cf2-c868-497b-8aa9-8ff3b60365f7
9	남양주 장사시설정보	남양주 장사시설정보	http://www.data.go.kr/dataset/fileDownload.do?atchFileId=FILE_000000001210050&fileDetailSn=1&publicDataDetailPk=uddi:082e3e8e-1609-4aec-97a5-c92a5db1d9b7

```
# [ URL이 기록된 csv를 읽어들여 다운로드 진행 ]

SAVE_DIR = 'C:/' # 저장 위치

def downloadURLResource(url): # URL 에서 파일 다운받는 함수 정의
    r = requests.get(url.rstrip(), stream=True)
    if r.status_code == 200:
        content_disposition = r.headers.get('content-disposition')
        if content_disposition is not None:
            targetFileName = requests.utils.unquote(cgi.parse_header(content_disposition)[1]['filename'])
            with open("{}/{}".format(SAVE_DIR, targetFileName), 'wb') as f:
                for chunk in r.iter_content(chunk_size=1024):
                    f.write(chunk)
            return targetFileName
        else: # 에러가 났을 경우 기록
            print('url {} had no content-disposition header'.format(url)) # content dispostion이 없는 경우
    elif r.status_code == 404:
        print('{} returned a 404, no file was downloaded'.format(url)) # r-status 코드가 404 인 경우
    else:
        print('something else went wrong with {}'.format(url)) # 기타 에러가 난 경우

with open('C:/') as f: # URL만 기록된 CSV 파일
    failItems = filter(lambda i:i[1] == False, {url.rstrip():downloadURLResource(url.rstrip()) for url in f.readlines()}.items())
    list(map(print, failItems))
```

다운로드 된 파일들의 형식

HWP CSV PDF XSLX PNG JPG
WMV MP4 MP3 Word PPT 외 11개

양식이 일정해 분석이 비교적 쉬운 CSV 파일 선택

01

02

03

04

1) CSV 파일 데이터 전처리

- 다운로드 한 CSV 파일에서 파일명과 필드 추출

```
# [ 다운받은 폴더에서 확장자가 csv 인 파일들에 대해 파일명과 필드 추출후 새 csv 파일 작성 ] - csv 파일준비

# *** 코드를 파일이 있는 폴더안에서 실행***

import csv
import glob
import os
lst=[]
files=glob.glob('C:/*.csv') # 유형이 '.csv' 인 파일
with open('C:/','w',encoding='cp949',newline='') as testfile: #새로운 csv 파일 생성
    csv_writer=csv.writer(testfile)
    for file in files:
        try:
            with open(file,'r') as infile:
                file=file[file.rfind('\\')+1:]
                reader=csv.reader(infile)
                headers=next(reader)
                headers=[str for str in headers if str]
                while len(headers) < 3 : # 받아온 행의 열이 3개 이하일 경우 필드명이 아니라고 가정. 다음 행 읽어오기
                    headers=next(reader)
                    headers=[str for str in headers if str]
                lst=[file]+headers
                csv_writer.writerow(lst)
        except:
            with open(file,'r',encoding='utf8') as infile: # 인코딩이 utf8로 된 파일의 경우
                file=file[file.rfind('\\')+1:]
                reader=csv.reader(infile)
                headers=next(reader)
                headers=[str for str in headers if str]
                headers[0] = headers[0].strip('\ufeff')
                while len(headers) < 3 :
                    headers=next(reader)
                    headers=[str for str in headers if str]
                lst=[file]+headers
                csv_writer.writerow(lst)
```

- 각 파일의 첫 행을 받아 오되, 열이 3개 미만일 경우 다음 행을 받도록 함 (3개 미만인 파일들은 별도로 검사)
- 파일의 인코딩이 cp949가 아닌 utf8 일 경우 읽는 과정에서 인코딩 지정
- 에러가 나는 77파일을 별도로 저장함

1) CSV 파일 데이터 전처리

	A	B	C	D	E	F	G	H
1	CSV 파일명	F1	F2	F3	F4	F5	F6	F7
2	06~'13친환경	사업년도	인증종류	신청농가수	신청면적	신청금액	지급농가수	지급면적
3	08~'10+토양	사업년도	선정년도	시도	시군	읍면동	비종	신청면적(r
4	11~'13+토양	사업년도	선정년도	시도	시군	읍면동	비종	신청면적(r
5	14~'16+토양	사업년도	선정년도	시도	시군	읍면동	비종	신청면적(r
6	14년+청년창	NO	창업자명	기업명	창업 내용	지역	기술분야	유형
7	14년+해외조	데이터ID	낙찰번호	낙찰 공고명	공고일	낙찰일	UNSPSC분	국가
8	14년+해외조	데이터ID	입찰번호	입찰공고명	마감일	공고일	기관명	국가
9	15년+글로벌	연번	비고	과정명	2015년 운	세 부 연 수 일 정		
10	15년+대구경	연번	비고	과정명	2015년 운	세 부 연 수 일 정		

	A	B	C	D
1	06~'13친환경농업직불제(인증종류별)			
2	08~'10+토양개량제지원+현황			
3	11~'13+토양개량제지원+현황			
4	14~'16+토양개량제지원+현황			
5	14년+청년창업사관학교+기업+현황			
6	14년+해외조달낙찰정보			
7	14년+해외조달입찰정보			
8	15년+글로벌리더십연수원+연수과정+목록			
9	15년+대구경북연수원+연수과정+목록			
10	15년+부산경남연수원+연수과정+목록			
11	15년+호남연수원+연수과정+목록			
12	(0)160104(공지)_2016년청년취업인턴제_운			
13	(141231)_2014_데이터_개방(전자상거래)			

14069 행

	A
1	사업년도
2	인증종류
3	신청농가수
4	신청면적
5	신청금액
6	지급농가수
7	지급면적
8	지급금액
9	사업년도
10	선정년도
11	시도
12	시군
13	읍면동
14	비종
15	신청면적(r
16	양(kg)

169464 행 / Null 값 41

```
# 필드명 일렬 나열시키는 excel 매크로 ( 필드 분석 시 필드 나열 파일 작성용,*** excel 에서 사용 *** )

Sub TableToColumn()
    Dim Rng As Range, LR As Long, i As Long
    LR = Range("B" & Rows.Count).End(xlUp).Row # B = 시작 열
    For i = 2 To LR
        Set Rng = Range("B" & i, "E" & i)  # B / E 대신 <- 시작/종료열
        Range("A" & Rows.Count).End(xlUp)(2).Resize(Rng.Count) = Application.WorksheetFunction.Transpose(Rng) # A <- 작성열
    Next i
End Sub
```

CSV 파일 데이터의 파일명과 필드명 분석을 위해 파일을 나눔.

2) OpenAPI 데이터 전처리

API명	API유형	오퍼레이션명	구분	항목명(영문)	항목명(국문)
기관조사가격서비스	REST	가락시장거래정보조회	요청메시지	from_date	조회 시작일(년월일)
기관조사가격서비스	REST	가락시장거래정보조회	요청메시지	to_date	조회 종료일(년월일)
기관조사가격서비스	REST	가락시장거래정보조회	요청메시지	lclass_name	부류(대품목)명
기관조사가격서비스	REST	가락시장거래정보조회	요청메시지	mclass_name	품목(중품목)명
기관조사가격서비스	REST	가락시장거래정보조회	요청메시지	sclass_name	품종(소품목)명
기관조사가격서비스	REST	가락시장거래정보조회	응답메시지	maxprice	최고가
기관조사가격서비스	REST	가락시장거래정보조회	응답메시지	mclassName	품목
기관조사가격서비스	REST	가락시장거래정보조회	응답메시지	minprice	최저가
기관조사가격서비스	REST	가락시장거래정보조회	응답메시지	sclassName	품종
기관조사가격서비스	REST	가락시장거래정보조회	응답메시지	sclasscode	품종코드

항목명이 열이 아닌 행으로 나열 된 상태

```
# [ Open API 데이터 정리 ] - 행마다 API 명이 반복되고 항목명만 다른 행들을 API 명 당 하나의 행으로 (API명+항목명) 정리

with open('C:/','w',encoding='cp949',newline='') as testfile:
    csv_writer=csv.writer(testfile)
    with open('C:/', 'r') as f:
        reader = csv.reader(f)
        for key, group in itertools.groupby(reader, lambda i:i[0]):
            lst=([key]+list(map(lambda i:i[1], group)))
            csv_writer.writerow(lst)
```

Python itertools 모듈 사용하여 코드 작성

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	API명	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
2	기관조사가	조회 시작	조회 종료	부류(대품	품목(중품	품종(소품	최고가	품목	최저가	품종	품종코드	규격코드	규격
3	지역간노선	(NULL)	출발지의	도착지의	(NULL)	운행노선	노선아이디	노선번호	노선유형	노선유형	노선의 운	노선의 관할지역 (서	
4	성인 검색	(NULL)											
5	블로그 AP	(NULL)											
6	수입쇠고기	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)				
7	상품이미지	(NULL)											
8	글 관련 AI	(NULL)											
9	노인사회활	한 페이지	페이지 번	시도/시군	시도	시군구	결과코드	결과메시지	한페이지	페이지 번	전체 결과	시도코드	시도명

정상적으로 하나의 행 마다 API 와 필드 나열

2) OpenAPI 데이터 전처리

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	API명	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16
2	기관조사가	조회 시작	조회 종료	부류(대품	품목(중품	품종(소품	최고가	품목	최저가	품종	품종코드	규격코드	규격	결과코드	결과메시지	한 페이지	페이지 번
3	지역간노선	(NULL)	출발지의	도착지의	(NULL)	운행노선목	노선아이디	노선번호	노선유형	노선유형	노선의 운	노선의 관할지역 (서울,경기,인천)					
4	성인 검색(	(NULL)															
5	블로그 AP	(NULL)															
6	수입쇠고기	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)								
7	상품이미지	(NULL)															
8	글 관련 AI	(NULL)															
9	노인사회활	한 페이지	페이지 번	시도/시군	시도	시군구	결과코드	결과메시지	한페이지	페이지 번	전체 결과	시도코드	시도명	시군구코드	시군구명	한 페이지	페이지 번

	A	B	C	D
1	기관조사가격서비스			
2	지역간노선 검색 서비스			
3	성인 검색어 판별			
4	블로그 API			
5	수입쇠고기 거래내역정보 등록 서비스			
6	상품이미지검색api			
7	글 관련 API			
8	노인사회활동(노인일자리) 시스템 코드 정보			
9	인빌쇼핑정보조회서비스			
10	관광실태조사서비스			
11	승강장정보서비스			
12	영문생활법령정보 조회 서비스			
13	테마체험관-전체			

2893 행

	A	B
1	조회 시작일(년월일)	
2	조회 종료일(년월일)	
3	부류(대품목)명	
4	품목(중품목)명	
5	품종(소품목)명	
6	최고가	
7	품목	
8	최저가	
9	품종	
10	품종코드	
11	규격코드	
12	규격	
13	결과코드	

81057 행 / Null 값 14755

항목명은 영문 / 국문으로 제공

국문 항목명만 없는 경우 14755 건 (필드값없음 12267 + (Null)로 기록 2488 개)

영문 항목명만 없는 경우 2879 건

영/국문 모두 없는 경우 : 2865 건

3) 표준 데이터 전처리

	A	B	C	D	E
1	name	category	f1	f2	f3
2	전국초중등학교위치표준	교육	학교ID	학교명	학교급구분
3	전국초등학교통학구역표	교육	학구ID	학구명	학구분류
4	전국고등학교학교군표준	교육	학구ID	학구명	학구분류
5	전국고등학교비평준화지	교육	학구ID	학구명	학구분류
6	전국교육행정구역표준데	교육	교육행정	교육행정구역	교육행정구역
7	전국학교학구도연계정보	교육	학구ID	학교ID	학교명
8	전국민방위대피시설표준	재난안전	민방위대	민방위대피시	소재지도로명
9	전국중학교학교군표준데	교육	학구ID	학구명	학구분류
10	전국야영(캠핑)장표준데이	문화관광	야영(캠핑	야영(캠핑)장-	위도

	A	B	C	D
1	전국초중등학교위치표준데이터			
2	전국초등학교통학구역표준데이터			
3	전국고등학교학교군표준데이터			
4	전국고등학교비평준화지역표준데이터			
5	전국교육행정구역표준데이터			
6	전국학교학구도연계정보표준데이터			
7	전국민방위대피시설표준데이터			
8	전국중학교학교군표준데이터			
9	전국야영(캠핑)장표준데이터			
10	전국치매센터표준데이터			
11	전국가로수길정보표준데이터			
12	전국건강증진센터표준데이터			
13	전국박물관미술관정보표준데이터			
14	전국농기계임대정보표준데이터			

	A
1	필드명
2	학교ID
3	학교명
4	학교급구분
5	설립일자
6	설립형태
7	본교분교구분

별도의 데이터 전처리 작업이 필요 없이 정리가 됨

파일 : 46

필드 : 884

1) 파일 별 필드수 기록

```
# [ 다운로드 완료된 CSV 파일에 대한 필드 수 기록 - 저장된 폴더에서 읽어오기 ]

files=glob.glob('C:/') #원본파일
with open('C:/','w',encoding='cp949',newline='') as testfile: #작성파일
    csv_writer=csv.writer(testfile)
    for file in files:
        try:
            with open(file,'r') as infile:
                file=file[file.rfind('₩₩')+1:]
                reader=csv.reader(infile)
                headers=next(reader)
                headers=[str for str in headers if str]
                while len(headers) < 3 :
                    headers=next(reader)
                    headers=[str for str in headers if str]
                lst=[file]+[len(headers)]
                csv_writer.writerow(lst)
        except:
            with open(file,'r',encoding='utf8') as infile:
                file=file[file.rfind('₩₩')+1:]
                reader=csv.reader(infile)
                headers=next(reader)
                headers=[str for str in headers if str]
                headers[0] = headers[0].strip('₩ufeff')
                while len(headers) < 3 :
                    headers=next(reader)
                    headers=[str for str in headers if str]
                lst=[file]+[len(headers)]
                csv_writer.writerow(lst)
```

CSV 데이터 파일 별 필드수 기록 코드

```
# [ 파일명과 필드 추출된 새 CSV 파일에서 필드 수 기록 ]

file1 = ('C:/') # 읽을 파일
file2 = ('C:/') # 작성 파일
with open(file1, 'r') as f1, open(file2, 'w', encoding='cp949', newline='') as f2:
    csv_reader = csv.reader(f1)
    csv_writer = csv.writer(f2)
    for row in csv_reader:
        x=[x for x in row if x]
        csv_writer.writerow([len(x)-1]) #파일명은 빼고 필드수만 기록하기 위해 -1
```

OpenAPI / 표준데이터 파일 별 필드수 기록 코드

2) 파일명 or 필드명 글자 길이 / 띄어쓰기 / 특수문자

```
def count_letters(word): # 글자 수를 세는 함수.
    BAD_LETTERS=["","  ","₩n","    ","    ","    ","     " ] # 제외할 글자를 명시
    return len([letter for letter in word if letter not in BAD_LETTERS])

file = "C:/" #원본파일
with open("C:/",'w',encoding='cp949',newline='') as testfile: #작성파일
    csv_writer=csv.writer(testfile)
    with open(file,'r') as fi:
            for each in fi:
                file=each
                linecount=count_letters(file)
                lst=[file]+[linecount]
                csv_writer.writerow(lst)
```

```
file = "C:/" #원본파일
with open("C:/",'w',encoding='cp949',newline='') as testfile: #작성파일
    csv_writer=csv.writer(testfile)
    with open(file,'r') as fi:
            for line in fi:
                file=line
                linecount=line.count(' ')
                lst=[file]+[linecount]
                csv_writer.writerow(lst)
```

```
regex = "[^가-힣a-zA-Z0-9₩n ]" # " " 를 제외한 모든 문자를 특수문자로 취급함.
list1=[]
file="C:/"
with open("C:/",'w',encoding='cp949',newline='') as testfile: #작성파일
    csv_writer=csv.writer(testfile)
    with open(file,'r') as fi:
            for line in fi:
                search_target = line
                result=re.findall(regex,search_target)
                if result != []:
                    list1=[line]+['Yes']+[len(result)]+result
                else :
                    list1=[line]+['No']+[len(result)]
                csv_writer.writerow(list1)
```

해당 코드 실행 시 값을 받아오지만,
탭/엔터의 유무에 따라 행이 늘어남

3) 파일명 or 필드명 글자 길이 / 띄어쓰기

```
# [ 글자 길이 세기 ]
# 행은 늘어나지 않으나 특정한 경우 열로 늘어남 ( 빈도 적음 ).

input_fileName = "C:/" #원본파일
output_fileName = "C:/" #출력파일

f = open(input_fileName, 'r')
out_list = []
buf = ''
flg = 0
for line in f:
    if line.count('"')%2 == 1:
        if flg == 0: flg = 1
        else: flg = 0
    if flg == 1: buf += line.strip('₩n')
    elif flg == 0 and len(buf) > 0:
        buf += line.strip('₩n')
        buf = buf.strip('"')
        out_list.append([buf,len(buf)])
        buf = ''
    else:
        line = line.strip('₩n')
        out_list.append([line,len(line)])
f.close()

of = open(output_fileName, 'w')
for each in out_list:
    print(each[0]+','+str(each[1]), file=of)
of.close()
```

```
# [ 띄어쓰기 길이 ]
# 행은 늘어나지 않으나 특정한 경우 열로 늘어남 ( 빈도 적음 ).

input_fileName = "C:/" #원본파일
output_fileName = "C:/" #출력파일

f = open(input_fileName, 'r')
out_list = []
buf = ' '
flg = 0
for line in f:
    if line.count('"')%2 == 1:
        if flg == 0: flg = 1
        else: flg = 0
    if flg == 1: buf += line.strip('₩n')
    elif flg == 0 and len(buf) > 0:
        buf += line.strip('₩n')
        buf = buf.strip('"')
        result=buf.count(' ')
        out_list.append([buf,result])
        buf = ''
    else:
        line = line.strip('₩n')
        result=line.count(' ')
        out_list.append([line,result])
f.close()

of = open(output_fileName, 'w')
for each in out_list:
    print(each[0]+','+str(each[1]), file=of)
of.close()
```

4) 파일명 or 필드명 글자 특수문자 여부 / 개수 세기

```
# [ 특수문자 여부 ]
# 행은 늘어나지 않으나 특정한 경우 열로 늘어남 ( 빈도 적음 ).

input_fileName = "C:/" #원본파일
output_fileName = "C:/" #출력파일

regex = "[^가-힣a-zA-Z0-9₩n ]"

f = open(input_fileName, 'r')
out_list = []
buf = ' '
flg = 0
for line in f:
    if line.count('"')%2 == 1:
        if flg == 0: flg = 1
        else: flg = 0
    if flg == 1: buf += line.strip(' ₩n')
    elif flg == 0 and len(buf) > 0:
        buf += line.strip(' ₩n')
        buf = buf.strip(' "')
        search_target=buf
        result=re.findall(regex,search_target)
        if result !=[]:
            result='Yes'
        else:
            result='No'
        out_list.append([buf,result])
        buf = ''
    else:
        line = line.strip(' ₩n')
        search_target=line
        result=re.findall(regex,search_target)
        if result !=[]:
            result='Yes'
        else:
            result='No'
        out_list.append([line,result])
f.close()

of = open(output_fileName, 'w')
for each in out_list:
    print(each[0]+','+str(each[1]), file=of)
of.close()
```

```
# elif 부분 아래 코드로 대체시 특수문자 개수 세기

elif flg == 0 and len(buf) > 0:
        buf += line.strip(' ₩n')
        buf = buf.strip(' "')
        search_target=buf
        result=re.findall(regex,search_target)
        result=[len(result)]
        out_list.append([buf,result])
        buf = ''
    else:
        line = line.strip(' ₩n')
        search_target=line
        result=re.findall(regex,search_target)
        result=[len(result)]
        out_list.append([line,result])
```

5) 결과 정리

	A	B	C	D	E	F	G	H	I	J	K
1		파일수	파일명			파일명 특수문자			파일명 띄어쓰기		
2			AVG	Max	Min	Yes	Max	Min	Yes	Max	Min
3	CSV	14069	22.156	63	2	13675	18	1	76	7	1
4	API	2893	11.608	31	1	425	7	1	2289	7	1
5	표준	46	12.565	17	10	2	2	1	0	0	0

6		필드수(기	파일별 필드수			필드명			필드명 특수문자			필드명 띄어쓰기		
7			AVG	Max	Min	AVG	Max	Min	Yes	Max	Min	Yes	Max	Min
8	CSV	169464/16	12.0445	233	1	5.811	53	1	27876	13	1	17374	9	1
9	API	81057/663	28.018/22.	1251	1	10.225	257	1	14342	72	1	26366	85	1
10	표준	884	19.217	69	9	5.772	17	1	48	4	1	0	0	0

- 특수문자와 띄어쓰기가 적을 수록 정리가 잘 되있는 파일이라고 판단이 가능.
- 표준 데이터의 경우 제공자가 공공데이터활용지원센터로 하나의 기관이 파일을 제공하기에 양식이 통일되어 정리된 모습을 확인 할 수 있음.
- CSV 데이터의 경우 파일별로 제공자가 다르며, 원래 CSV 파일이 아닌 파일들을 CSV 로 옮겨 오면서 값들 사이에 특수문자 / 띄어쓰기가 생겨 버린 모습을 확인 할 수 있음.

1) 추후 계획 및 기대 효과

파일/필드명 클러스터링을 통한 정리

파일/필드 데이터 이외에 실제 데이터들에 대한 분석

데이터와 코드 공개를 통해 일반 시민 참여 유도

필드 간 관계 분석을 통한 데이터셋 간의 새로운 관계 구축 가능

공공데이터 포탈 관리자에 제공 데이터 수정의 필요성 제고, 궁극적으로는 데이터의 양식이 통일 되며 파일 오류가 줄어들어 공공데이터 사용자로 하여금 전처리에 소요되는 시간을 줄어 들 수 있게 함.

감사합니다