데이터 분석

1. 그래프 작성
2. 빈도 분석: 문장 형태소 분석 – KoNLPy
3. 명사 추출 시각화 :
4. 데이터기반 추출 서비스-상관관계 분석,pandas, network
5. 지도 시각화-Folium

1. 그래프 작성

- 기본 그래프 그리기

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4])
plt.xlabel('X-axis label')
plt.ylabel('Y-axis label')
plt.show()
```

```
plt.plot([1,2,3,4],[1,2,3,4])  # x 좌표가 1부터 시작
plt.show()
```

```
plt.plot([1,2,3,4], [1,2,3,4], 'ro')  # ro  -> bv 수정해보기
plt.show()
```

1. 그래프 작성

복수개의 시리즈 그리기

```
plt.plot([1,2,3,4],[1,2,3,4],'r-', [1,2,3,4],[3,4,5,6],'v-')
plt.show()
```

한글처리

```
plt.plot([1,2,3,4])
plt.xlabel('x축 한글표시')
plt.show()
```

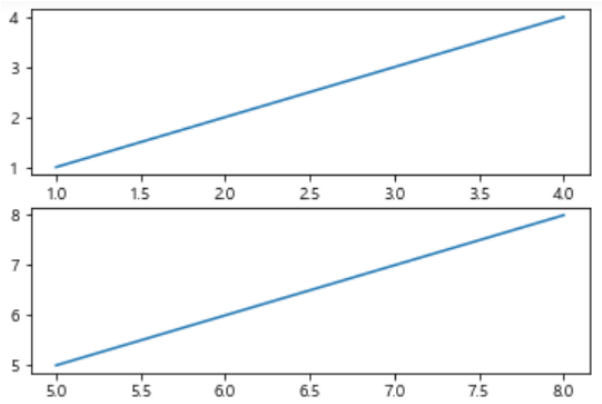
한글 깨짐

```
from matplotlib import font_manager, rc
import matplotlib
font_location = "c:/Windows/fonts/malgun.ttf"
font_name = font_manager.FontProperties(fname=font_location).get_name()
matplotlib.rc('font', family=font_name)
plt.plot([1,2,3,4])
plt.xlabel('x축 한글표시')
plt.show()
```

1. 그래프 작성

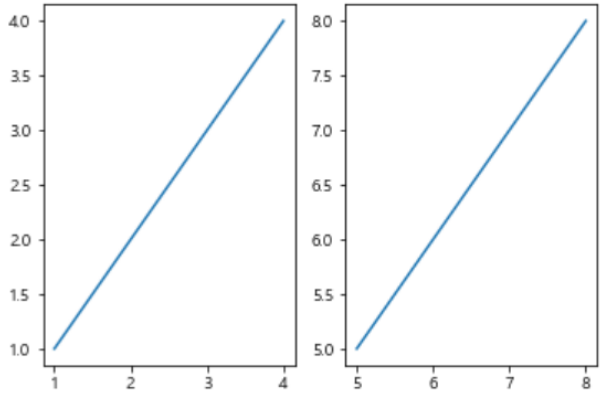
▫ 한 화면에 여러 개 그래프 그리기

```
plt.figure()
plt.subplot(2,1,1)
plt.plot([1,2,3,4], [1,2,3,4])
plt.subplot(2,1,2))
plt.plot([5,6,7,8],[5,6,7,8])
plt.show()
```



figure()
- 하나의 캔버스 생성
subplot(m,n,idx)
- 캔버스에 여러 개 그림 넣기
m : 행수
n : 열수
idx : 위치

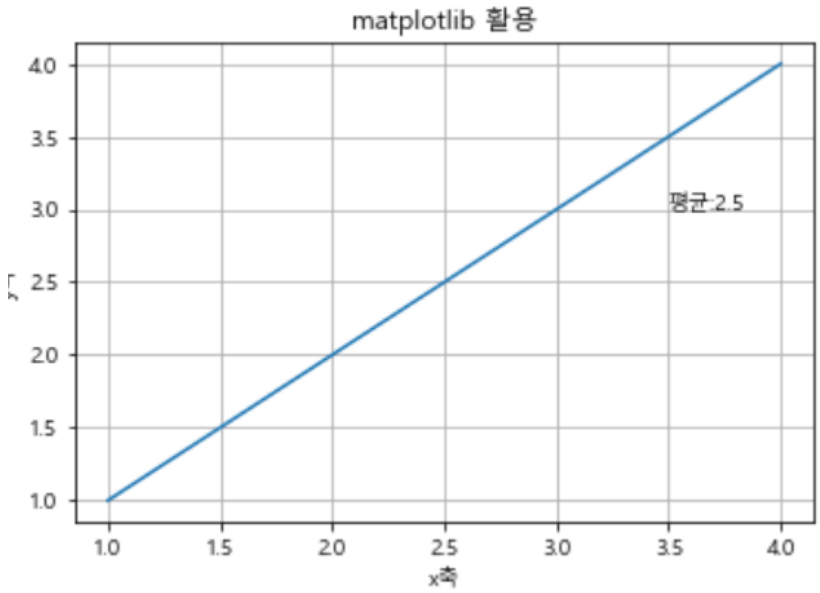
```
plt.figure()
plt.subplot(1, 2, 1)
plt.plot([1,2,3,4], [1,2,3,4])
plt.subplot(1,2,2)
plt.plot([5,6,7,8],[5,6,7,8])
plt.show()
```



1. 그래프 작성

- 그래프에 문자 삽입

```
plt.plot([1,2,3,4], [1,2,3,4])
plt.xlabel('x축')
plt.ylabel('y축')
plt.title('matplotlib 활용')  # 그래프 제목
plt.text(3.5, 3.0, '평균:2.5') #그래프의 지정위치에 문자열 표시
plt.grid(True) # 인덱스마다 격자를 나타냄
plt.show()
```



2. 빈도 분석: 문장 형태소 분석 – KoNLPy

- 문장 형태소 분석
  - 어떤 대상 어절의 모든 가능한 분석 결과를 출력하는 것을 의미
  - 효율적 정보 검색을 위해서 문장중 주요 단어(색인어) 추출
    - 문장중에 '명사'를 추출
  - 한글 명사 추출 어려움
    - : 조사와 복합명사(명사 두개가 합쳐져 만들어진 명사)등으로 영어권 문장에 비하여 추출이 어려움

복합 명사의 예	"논밭" : '논' + '밭' 또는 "논밭" "눈물" : '눈' + '물' 또는 "눈물"
조사의 예	"나는" : '나'(대명사) + '는'(조사)
어미의 예	"나는" : '나'(동사: 날다) + '는'(관형형 어미)

2. 빈도 분석: 문장 형태소 분석 – KoNLPy

- 코엔엘파이(KoNLPy)의 설치 및 활용
  - 참고 문헌 : http://konlpy-ko.readthedocs.io/ko/v0.4.3/install
  - 설치과정
    - JAVA 1.7 이상의 설치
    - JAVA_HOME Path 설정
    - JPype1 (>=0.5.7) 설치
  - 관리자권한으로 conda prompt실행

```
conda install -c conda-forge jpype1
```

  - 관리자권한으로 명령 프롬프트 실행

```
pip install konlpy
```

- **품사 태깅(tag)**
  - 형태소의 뜻과 문맥을 고려하여 그것에 마크업을 하는 것
  - **예 :** 가방에 들어가신다 -> 가방/NNG(일반명사) + 에/JKM(부사격조사) + 들어가/VV(동사) + 시/EPH(존칭선어말 어미) + ㄴ다/EFN(평서형 종결의미)
  - 한국어 품사 태그 비교표 링크: https://docs.google.com/spreadsheets/d/1OGAjUvalBuX-oZvZ_-9tEfYD2gQe7hTGsgUpiiBSXI8/edit#gid=0

- **말뭉치(corpus)**
  - Hannanum - KAIST 말뭉치를 이용해 생성된 사전
  - Kkma - 세종 말뭉치를 이용해 생성된 사전 (꼬꼬마)
  - Mecab - 세종 말뭉치로 만들어진 CSV형태의 사전
  - Komoran- Java로 쓰여진 오픈소스 한글 형태소 분석기
  - Twitter(Okt) - 오픈소스 한글 형태소 분석기

Hannanum 실습

```
from konlpy.tag import Hannanum
hannanum = Hannanum()

hannanum.analyze   #구(Phrase) 분석
hannanum.morphs    #형태소 분석
hannanum.nouns     #명사 분석
hannanum.pos       #형태소 분석 태깅

# 사용예시
print(hannanum.analyze(u'롯데마트의 흑마늘 양념 치킨이 논란이 되고 있다.'))

print(hannanum.morphs(u'롯데마트의 흑마늘 양념 치킨이 논란이 되고 있다.'))

print(hannanum.nouns(u'다람쥐 헌 쳇바퀴에 타고파'))

print(hannanum.pos(u'웃으면 더 행복합니다!'))
```

Kkma 실습

```
from konlpy.tag import Kkma
kkma = Kkma()

kkma.morphs          #형태소 분석
kkma.nouns           #명사 분석
kkma.pos             #형태소 분석 태깅
kkma.sentences       #문장 분석

# 사용예시
print(kkma.morphs(u'공부를 하면할수록 모르는게 많다는 것을 알게 됩니다.'))

print(kkma.nouns(u'대학에서 DB, 통계학, 이산수학 등을 배웠지만...'))

print(kkma.pos(u'다 까먹어버렸네요?ㅋㅋ'))

print(kkma.sentences(u'그래도 계속 공부합니다. 재밌으니까!'))
```

Okt 실습

```
from konlpy.tag import Okt
okt = Okt()

okt.morphs      #형태소 분석
okt.nouns       #명사 분석
okt.phrases     #구(Phrase) 분석
okt.pos         #형태소 분석 태깅

# 사용예시
print(okt.morphs(u'단독입찰보다 복수입찰의 경우'))

print(okt.nouns(u'유일하게 항공기 체계 종합개발 경험을 갖고 있는 KAI는'))

print(okt.phrases(u'날카로운 분석과 신뢰감 있는 진행으로'))

print(okt.pos(u'이것도 되나욬ㅋㅋ'))
```

Konlpy.utils 클래스에서 유용한 메서드

```
konlpy.utils.concordance(찾을 단어, 전체문장, show=False)
#concordance 단어색인(해당 단어가 쓰인 부분(index) 찾음)
#show = True 인 경우 단어 색인과 함께 포함된 문장 반환

konlpy.utils.pprint(obj) #유니코드 문자 출력

konlpy.utils.read_json(filename, encoding=u'utf-8') # json 파일 읽음
konlpy.utils.read_txt(filename, encoding=u'utf-8')    # txt 파일 읽음
konlpy.utils.csvread(f, encoding=u'utf-8')                  # csv 파일 읽음
konlpy.utils.csvwrite(data, f)                          # csv파일로 쓰기 가능
konlpy.utils.hex2char( h )      #16진수 문자를 유니코드 문자로 변환
```

3. 다빈도 명사추출 시각화

- **사용 모듈**
  - KoNLPy
  - matplotlib
  - PyTagCloud
  - WordCloud

3. 다빈도 명사추출 시각화

- **WordCloud 그리기-pytagcloud**

```
from collections import Counter # 단어들을 집계하기 위해서 사용
from konlpy.tag import Okt  # 형태소 분석기
import pytagcloud  #pygame 패키지에 의존적, pygame 설치 요구
# pip install pytagcloud
```

```
f=open("data_al/data.txt",encoding='utf-8')
data=f.read()
#data
npl=Okt()  # 형태소 분석기 Okt 생성,Kkma, Mecab-kr, HanNum 등
nouns=npl.nouns(data) # 형태소 분석기로 단어 추출
#nouns
count=Counter(nouns) # 단어 집계
#count
tag2=count.most_common(20)  # 상위 20개만 추출
taglist=pytagcloud.make_tags(tag2,maxsize=80)  # tag2데이터로 태그 생성
#taglist
f.close()
```

3. 다빈도 명사추출 시각화

- **WordCloud 그리기-pytagcloud**
  - pytagcloud.create_tag_image() 메서드
    - 워드클라우드를 이미지로 출력
    - 매개 변수
      - 태그목록
      - 저장될 파일명
      - size : 캔버스 사이즈. 다음과 같이 명시합니다. size=(640,480)
      - fontname : 글꼴 종류. 기본적으로 지원하는 글꼴의 종류

        Nobile, Old Standard TT, Cantarell, Reenie Beanie, Cuprum, Molengo, Neucha, Philosopher, Yanone Kaffeesatz, Cardo, Neuton, Inconsolata, Crimson Text, Josefin Sans, Droid Sans, Lobster, IM Fell DW Pica, Vollkorn, Tangerine, Coustard, PT Sans Regular
      - rectangular : 워드클라우드를 사각형박스 형태로 할 건지, 원형으로 출력할 건지 명시

```
create_tag_image(taglist, 'wordcloud.jpg', size=(900, 600),
fontname='Nobile', rectangular=False)
```

이미지 확인하면 한글 깨짐

3. 다빈도 명사추출 시각화

- **WordCloud 그리기-pytagcloud**
  - 한글 글꼴 설정
    - C:/anaconda3/Lib/site-packages/pytagcloud/fonts/fonts.json 열어서 다음 내용을 추가
    - C:/anaconda3/Lib/site-packages/pytagcloud/fonts에 NanumBarunGothic.ttf 파일을 다운받아 넣어준다.

```
[
        {
                "name": "korean",
                "ttf": "NanumBarunGothic.ttf",
                "web": "http://fonts.googleapis.com/css?family=Nobile"
        },
        {
                "name": "Nobile",
                "ttf": "nobile.ttf",
                "web": "http://fonts.googleapis.com/css?family=Nobile"
        },
        {
                "name": "Old Standard TT",
                "ttf": "OldStandard-Regular.ttf",
                "web": "http://fonts.googleapis.com/css?family=Old+Standard+TT"
        },
```

3. 다빈도 명사추출 시각화

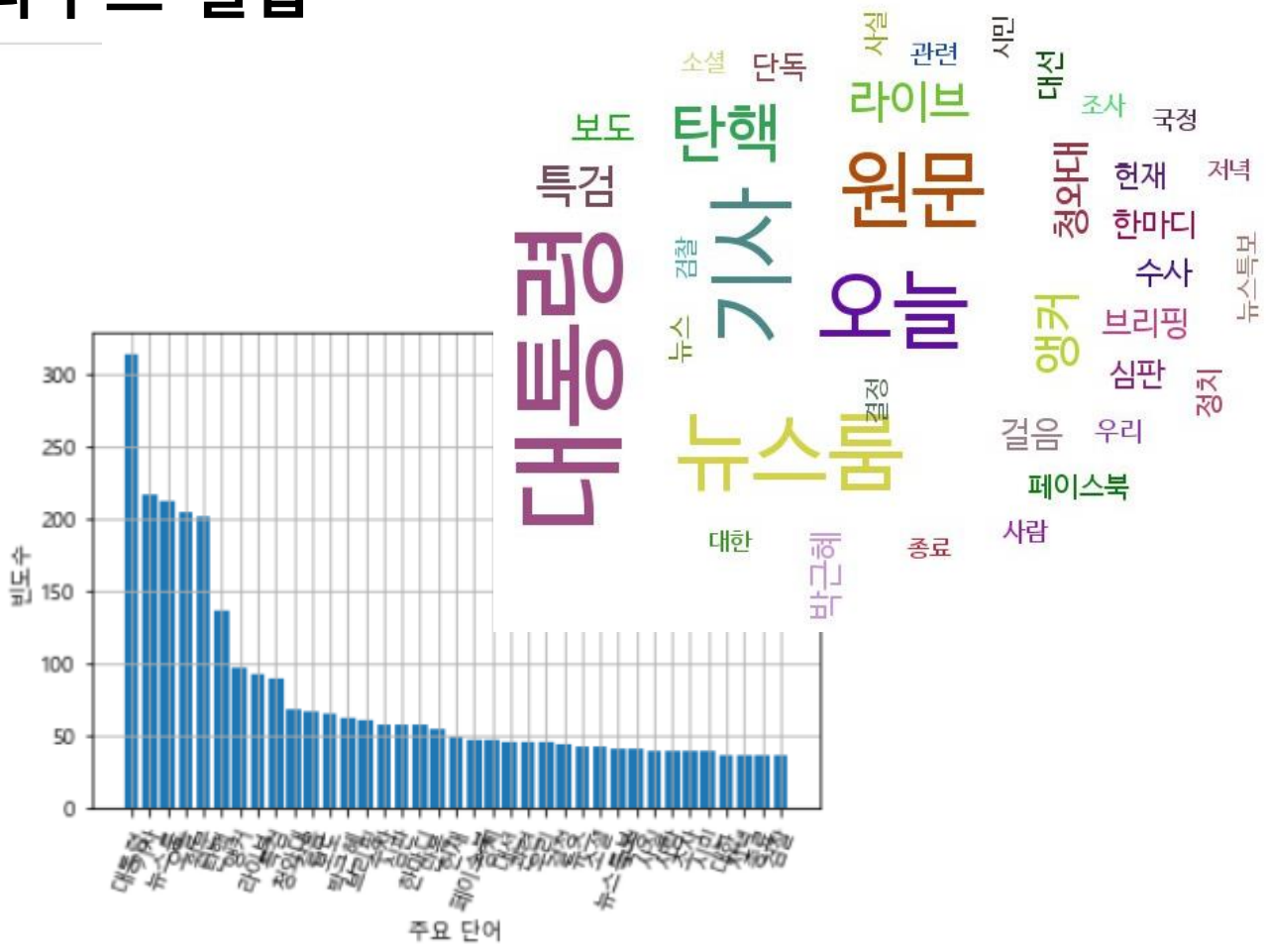
- **WordCloud 그리기-pytagcloud**
  - Kernel-> Restart
  - create_tag_image() 함수의 fontname='korean'으로 변경

```
pytagcloud.create_tag_image(taglist, 'wordcloud.jpg', size=(900, 600),
fontname='korean', rectangular=False)
```

3. 다빈도 명사추출 시각화

워드클라우드 실습

대통령 : 314
기사 : 217
뉴스룸 : 212
오늘 : 204
원문 : 202
탄핵 : 137
앵커 : 97
라이브 : 92
특검 : 90
청와대 : 69
걸음 : 67
보도 : 65
박근혜 : 62
브리핑 : 61
수사 : 58
심판 : 58
한마디 : 57
단독 : 54
헌재 : 49
페이스북 : 47
정치 : 47
대선 : 46
관련 : 45
우리 : 45
결정 : 44
뉴스 : 43
소셜 : 42
뉴스특보 : 41
국정 : 41
사실 : 40
사람 : 40
조사 : 40
시민 : 39
대한 : 37
저녁 : 37
종료 : 36
검찰 : 36



3. 다빈도 명사추출 시각화

- **pytagcloud : 실습**

```
import json
import re #정규표현식(regular expression)

from konlpy.tag import Okt
from collections import Counter

import matplotlib.pyplot as plt
import matplotlib
from matplotlib import font_manager, rc

import pytagcloud
import webbrowser
```

3. 다빈도 명사추출 시각화

#[CODE 1]

```
def showGraph(wordInfo):

    font_location = "c:/Windows/fonts/malgun.ttf"
    font_name = font_manager.FontProperties(fname=font_location).get_name()
    matplotlib.rc('font', family=font_name)

    plt.xlabel('주요 단어')
    plt.ylabel('빈도수')
    plt.grid(True)

    Sorted_Dict_Values = sorted(wordInfo.values(), reverse=True)
    Sorted_Dict_Keys = sorted(wordInfo, key=wordInfo.get, reverse=True)

    plt.bar(range(len(wordInfo)), Sorted_Dict_Values, align='center')
    plt.xticks(range(len(wordInfo)), list(Sorted_Dict_Keys), rotation='70')

    plt.show()
```

3. 다빈도 명사추출 시각화

```
#[CODE 2]

def saveWordCloud(wordInfo, filename):

    taglist = pytagcloud.make_tags(dict(wordInfo).items(), maxsize=80)
    pytagcloud.create_tag_image(taglist, filename, size=(640, 480),
            fontname='korean', rectangular=False)
    webbrowser.open(filename)
```

3. 다빈도 명사추출 시각화

```
def main():
    #여기서 파일의 경로는 실제 JSON 데이터가 저장된 경로이다

    openFileName = 'data_al/jtbcnews_facebook_2016-10-01_2017-03-12.json'
    #openFileName = 'd:/Temp/FB_DATA/jtbcnews_facebook_2016-10-01_2017-03-12.json'

    cloudImagePath = openFileName + '.jpg'

    rfile = open(openFileName, 'r', encoding='utf-8').read()

    jsonData = json.loads(rfile)
    message = ''
```

3. 다빈도 명사추출 시각화

```
#[CODE 3]

for item in jsonData:
    if 'message' in item.keys():
        message = message + re.sub(r'[^₩w]', ' ', item['message']) + ' '

#[CODE 4]

nlp = Okt()
nouns = nlp.nouns(message)
count = Counter(nouns)
…
```

3. 다빈도 명사추출 시각화

```
    ....
    #[CODE 5]

    wordInfo = dict()
    for tags, counts in count.most_common(50):
        if (len(str(tags)) > 1):
            wordInfo[tags] = counts
            print ("%s : %d" % (tags, counts))

    showGraph(wordInfo)
    saveWordCloud(wordInfo, cloudImagePath)

if __name__ == "__main__":
    main()
```

3. 다빈도 명사추출 시각화

- **pytagcloud : 실습**

```
["삼성 임원진들 준법서약 · · · '뉴삼성' 도약 시동", "삼성전자 주가 상승에 삼성그룹주 펀드도 '신바람'", '"7%p차 화웨이 추격" 삼성전자, 글로벌 통신장비시장 광폭 행보(종합)', '[이봉수의 참!]삼성은 언론과 국민을 바보로 만드나', '연일 새 역사… 삼성전자 주가 6만원 뚫었다', "삼성전자, 美 5G망 설계 전문 '텔레웰드 솔루션즈' 인수", '삼성 최고가 행진… '6만 전자' 굳히고, '7만 전자' 가나', "[르포] 베이스를 뚫고 나오는 목소리… '삼성 사운드 기술의 산실' 가 보니", '삼성전자, 14년 연속 美 특허 2위… 韓 기업 11개사 '톱 100' ', '삼성그룹 "위법행위 묵과 않겠다"', "삼성전자-SK하이닉스 경영진 '대박'…웃지 못하는 CEO", '삼성전자 사장단 "준법 경영" 서약', '삼성전자, 준법실천 서약식 개최…사장단 직접 서명', '"화웨이는 가라"…삼성 #'갤럭시Z 플립'#으로 모토로라와 #'진검승부'#', '삼성전자 · SK하이닉스, 장초반 또 신고가 경신', '삼성 새 폴더블폰 '갤럭시Z 플립' 어떻게 생겼나', '삼성현대차SKLG · 롯데포스코, 재계서열 1~6위 10년 간 그대로', 'KT-삼성서울병원, 5G 도입해 의료 정보 실시간 공유', '갤S10 후속작은 갤S20… 삼성 신작 사양 잇따라 유출', '삼성 차세대 폴더블폰 모습은?…"가로로 접는 방식"', '삼성전자 준법실천 서약식', "KT-삼성서울병원, 5G로 '스마트 병원' 만든다", '삼성전자, 5G 공략 강화…미국 네트워크 망설계 기업 인수', '반올림 측 "국회 통과한 #'삼성보호법'#, 치밀한 기획으로 완성"', '[마켓뷰] 삼성전자 · SK하이닉스, 각각 6만원 · 10만원 뚫었다', 'KT, 삼성서울병원과 5G 스마트 혁신병원 만든다', "[단독] 검찰, 삼성 준법감시위에 '승계 관련 자료' 요청",
'삼성전자, 미국 5G · 4G 망설계 기업 텔레웰드솔루션즈 인수', ' '사상 최고치' 삼성전자 주식…1년새 51% 급
장기화되면 부산 경제 다 죽는다." ', "사라지는 檢 직접수사...'삼성 합병 의혹' 등 수사 차질 빚나", "삼성
```

```
[('삼성', 'Noun'), ('임원진', 'Noun'), ('들', 'Suffix'), ('준법', 'Noun'), ('서약', 'Noun'), ("· · ·
n'), ('삼성', 'Noun'), ("'", 'Punctuation'), ('도약', 'Noun'), ('시동', 'Noun')]
------------------------------
[('삼성', 'Noun'), ('전자', 'Noun'), ('주가', 'Noun'), ('상승', 'Noun'), ('에', 'Josa'), ('삼성', 'N
'Noun'), ('펀드', 'Noun'), ('도', 'Josa'), ("'", 'Punctuation'), ('신바람', 'Noun'), ("'", 'Punctuat
------------------------------
[('"', 'Punctuation'), ('7%', 'Number'), ('p', 'Alpha'), ('차', 'Noun'), ('화웨이', 'Noun'), ('추격'
('삼성', 'Noun'), ('전자', 'Noun'), (',', 'Punctuation'), ('글로벌', 'Noun'), ('통신', 'Noun'), ('장
('광폭', 'Noun'), ('행보', 'Noun'), ('(', 'Punctuation'), ('종합', 'Noun'), (')', 'Punctuation')]
------------------------------
```

```
[[('삼성', 'Noun'), ('임원진', 'Noun'), ('들', 'Suffix'), ('준법', 'Noun'), ('서약', 'Noun'), (" · · · '",
un'), ('삼성', 'Noun'), ("'", 'Punctuation'), ('도약', 'Noun'), ('시동', 'Noun')], [('삼성', 'Noun'), ('
un'), ('상승', 'Noun'), ('에', 'Josa'), ('삼성', 'Noun'), ('그룹', 'Noun'), ('주', 'Noun'), ('펀드', 'Nou
'Punctuation'), ('신바람', 'Noun'), ("'", 'Punctuation')], [('"', 'Punctuation'), ('7%', 'Number'), ('p',
('화웨이', 'Noun'), ('추격', 'Noun'), ('"', 'Punctuation'), ('삼성', 'Noun'), ('전자', 'Noun'), (',', 'Pu
un'), ('통신', 'Noun'), ('장비', 'Noun'), ('시장', 'Noun'), ('광폭', 'Noun'), ('행보', 'Noun'), ('(', 'Pu
n'), (')', 'Punctuation')], [('[', 'Punctuation'), ('이봉수', 'Noun'), ('의', 'Josa'), ('참', 'Verb'), ('
성', 'Noun'), ('은', 'Josa'), ('언론', 'Noun'), ('과', 'Josa'), ('국민', 'Noun'), ('을', 'Josa'), ('바보'
('만드', 'Noun'), ('나', 'Josa')], [('연일', 'Noun'), ('새', 'Noun'), ('역사', 'Noun'), ('…', 'Punctuati
('전자', 'Noun'), ('주가', 'Noun'), ('6만원', 'Number'), ('뚫었다', 'Verb')], [('삼성', 'Noun'), ('전자',
n'), ('美', 'Foreign'), ('5', 'Number'), ('G', 'Alpha'), ('망', 'Noun'), ('설계', 'Noun'), ('전문', 'Noun
('텔레', 'Noun'), ('월드', 'Noun'), ('솔루션', 'Noun'), ('즈', 'Modifier'), ("'", 'Punctuation'), ('인수
```


```
[('삼성', 765), ('전자', 443), ('병원', 171), ('준법', 158), ('서약', 110), ('실천', 100), ('서울', 93), ('인수', 89), ('하이닉스',
88), ('기업', 80), ('혁신', 78), ('스마트', 75), ('망', 72), ('설계', 68), ('주가', 57), ('개최', 51), ('최고', 47), ('경영', 43),
('갤럭시', 38), ('폰', 36), ('공략', 36), ('전문', 35), ('시장', 32), ('또', 31), ('경신', 31), ('의료', 30), ('감', 30), ('시위', 3
0), ('텔레', 29), ('월드', 29)]
```

3. 다빈도 명사추출 시각화

- **WordCloud 그리기-WordCloud**

```
%matplotlib inline
from bs4 import BeautifulSoup
import requests
from konlpy.tag import Okt
from collections import Counter
from wordcloud import WordCloud
import matplotlib.pyplot as plt

search_word = "삼성"  # 검색어 지정
title_list = []
```

3. 다빈도 명사추출 시각화

```
def get_titles(start_num, end_num):
    #start_num ~ end_num까지 크롤링
    while 1:
        if start_num > end_num:
            break
        print(start_num)

        url = 'https://search.naver.com/search.naver?where=news&sm=tab_jum
                        &query={}&start={}'.format(search_word,start_num)
        req = requests.get(url)

        if req.ok:  # 정상적인 request 확인
            html = req.text
            soup = BeautifulSoup(html, 'html.parser')
             # 뉴스제목 뽑아오기
            titles = soup.select('ul.type01 > li > dl > dt > a')

            # list에 넣어준다
            for title in titles:
                title_list.append(title['title'])
        start_num += 10
    #print(title_list)
```

3. 다빈도 명사추출 시각화

```
def make_wordcloud(word_count):
    okt = Okt()

    sentences_tag = []
    #형태소 분석하여 리스트에 넣기
    for sentence in title_list:
        morph = okt.pos(sentence)
        sentences_tag.append(morph)
        print(morph)
        print('-' * 30)

    print(sentences_tag)
    print('₩n' * 3)

    noun_adj_list = []
    #명사와 형용사만 구분하여 이스트에 넣기
    for sentence1 in sentences_tag:
        for word, tag in sentence1:
            if tag in ['Noun', 'Adjective']:
                noun_adj_list.append(word)
```

3. 다빈도 명사추출 시각화

```
#형태소별 count
    counts = Counter(noun_adj_list)
    tags = counts.most_common(word_count)
    #print(tags)

    #wordCloud생성
    #한글깨지는 문제 해결하기위해 font_path 지정
    wc = WordCloud(font_path='D:/fonts/NanumBarunGothic.ttf',
            background_color='white', width=800, height=600)
    #print(dict(tags))
    cloud = wc.generate_from_frequencies(dict(tags))
    plt.figure(figsize=(10, 8))
    plt.axis('off')
    plt.imshow(cloud)
    plt.show()

if __name__ == '__main__':
     get_titles(1,200) #1~200번게시글 까지 크롤링
     make_wordcloud(30) #단어 30개까지 wordcloud로 출력
```

4. 데이터 기반 추천 :데이터 상관관계 분석

상관분석과 상관계수

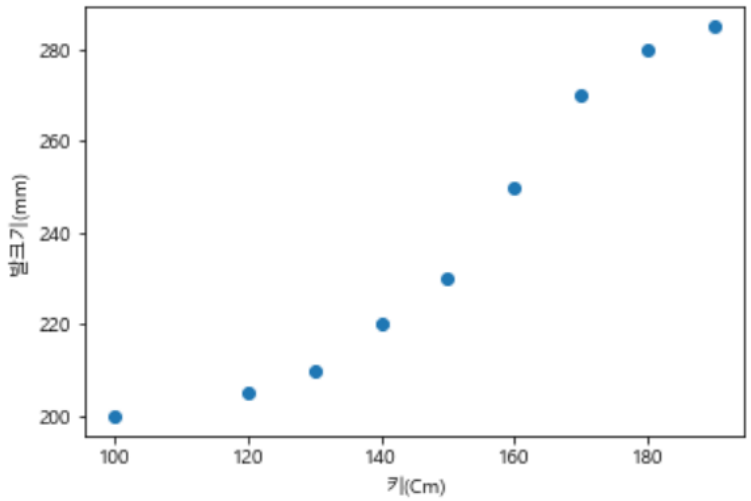
- 우리는 일상생활에서 “키가 크면 발이 크다”, “교육수준이 높을수록 자녀의 대학 진학률이 높다”등의 이야기를 많이 한다.
- 이러한 두 개의 변수 ‘키’와 ‘발’, ‘교육수준’과 ‘자녀 대학 진학률’간의 관계가 어떠한 관계를 가지고 있는지 성향을 분석하는 것이 “상관분석”이다.
- 이를 ‘산포도’ 또는 ‘산점도’라는 그래프로 그리면 직관적으로 두 변수간의 관계를 파악할 수 있다.
- 예) 어느 집단의 키와 발크기

키(Cm)	100	120	130	140	150	160	170	180	190
발크기(mm)	200	205	210	220	230	250	270	280	285

상관분석과 상관계수

산포도 그래프 작성

```
import matplotlib.pyplot as plt
from matplotlib import font_manager, rc
import matplotlib
font_location = "c:/Windows/fonts/malgun.ttf"
font_name = font_manager.FontProperties(fname=font_location).get_name()
matplotlib.rc('font', family=font_name)
height = [100, 120, 130, 140, 150, 160, 170, 180, 190]
foot_size = [200, 205, 210, 220, 230, 250, 270, 280, 285]
plt.scatter(height, foot_size)
plt.xlabel('키(Cm)')
plt.ylabel('발크기(mm)')
plt.show()
```

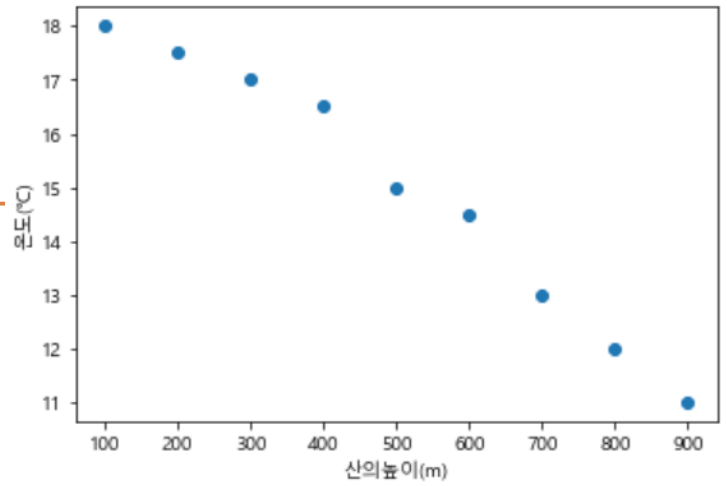


상관분석과 상관계수

- 예 : 산의 높이에 따른 온도의 변화
  - 일반적으로 산의 높이가 100m 올라갈때마다 평균 0.65℃ ~ 0.75℃ 정도 낮아진다고 한다.

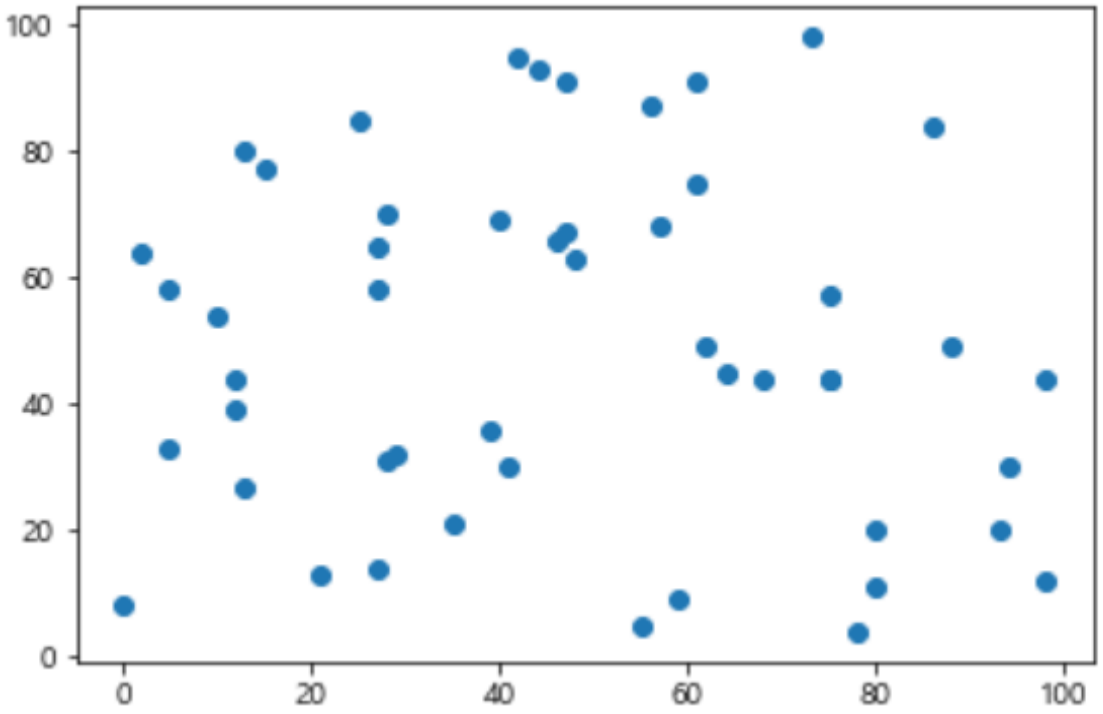
산의 높이(m)	100	200	300	400	500	600	700	800	900
온도(℃)	18.0	17.5	17	16.5	15	14.5	13	12	11

```
height = [100, 200, 300, 400, 500, 600, 700, 800, 900]
temperature = [18.0, 17.5, 17, 16.5, 15, 14.5, 13, 12, 11]
plt.scatter(height, temperature)
plt.xlabel('산의높이(m)')
plt.ylabel('온도(℃)')
plt.show()
```



상관분석과 상관계수

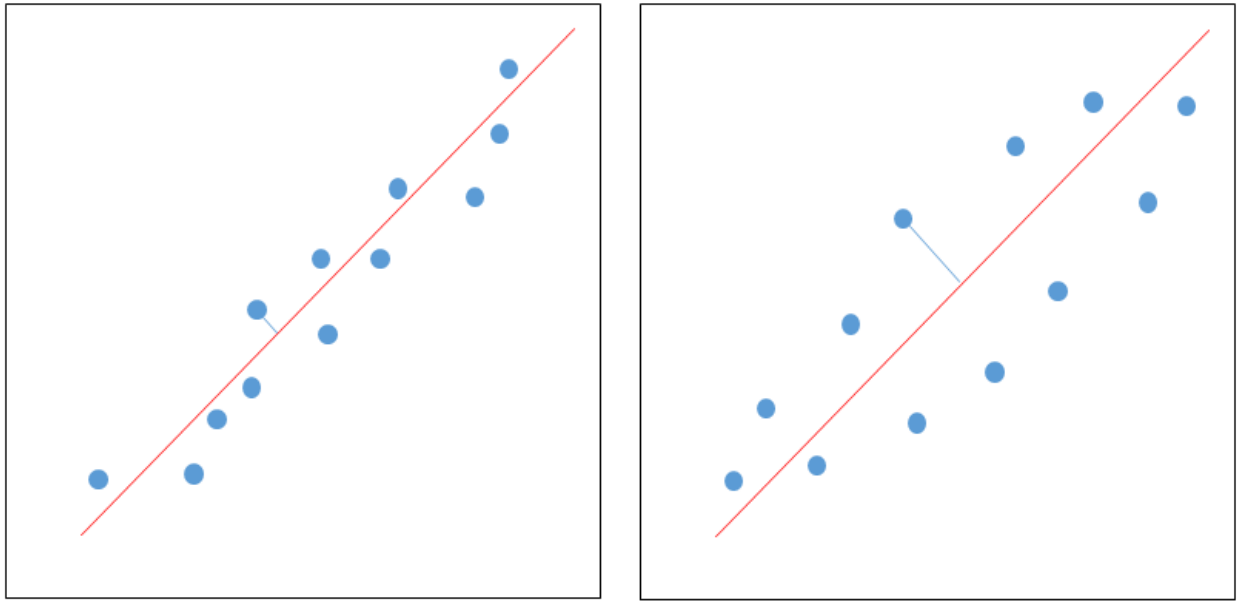
```
import numpy as np
random_x = np.random.randint(0, 100, 50)
random_y = np.random.randint(0, 100, 50)
plt.scatter(random_x, random_y)
plt.show()
```



상관분석과 상관계수

상관계수

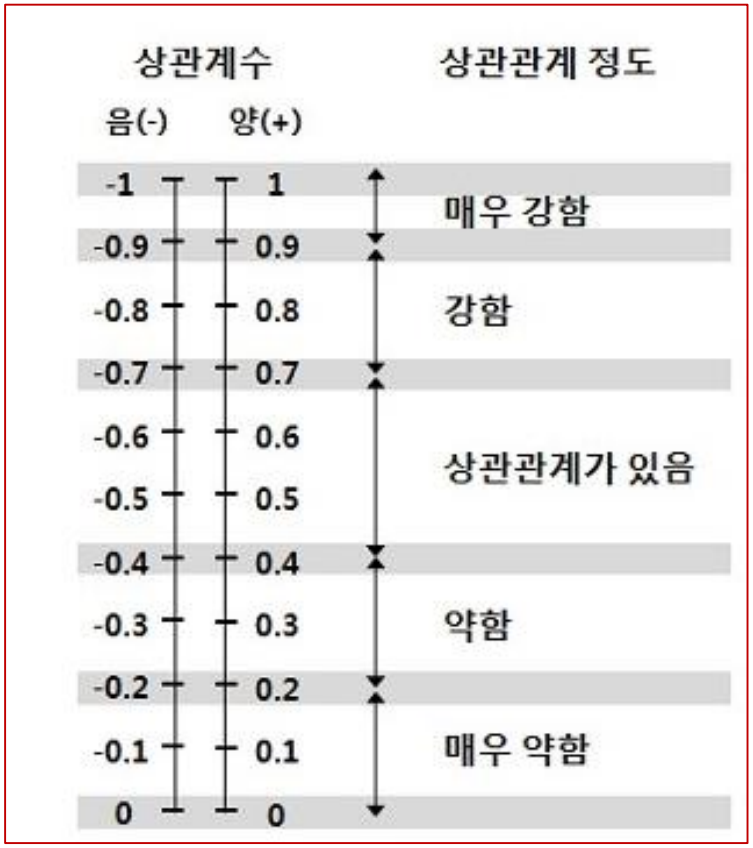
- 상관분석에서는 x의 값이 증가함에 따라 y의 값이 증가하는 경우 우리는 “양의 상관관계”를 가지고 있다.
- 상관 계수는 서로간의 데이터가 어느 정도의 근접도를 가지고 있는지 표현하는 방법



[그림 4] 양의 상관관계를 가지고 있으나 근접도가 다른 경우

상관분석과 상관계수

- 상관계수(기호 : r)
  - $-1 \le r \le 1$ 의 값을 가짐



- 상관관계 분석식

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

$$= \frac{\mathbf{E}[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

$$= \frac{\mathbf{E}(XY) - \mathbf{E}(X)\mathbf{E}(Y)}{\sigma_X \sigma_Y}$$

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E(X)^2}\sqrt{E(Y^2) - E(Y)^2}}.$$

모집단(전체)의 상관계수 공식

$$r = \frac{\sum_{i=1}^{n}(x-\overline{x})(y-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\overline{y})^2}}$$

표본(일부 샘플들)의 상관계수 공식

상관분석 –실습1

```
import math

def correlation(x, y):
    n = len(x)
    vals = range(n)
    x_sum = 0.0
    y_sum = 0.0
    x_sum_pow = 0.0
    y_sum_pow = 0.0
    mul_xy_sum = 0.0

    for i in vals:
        mul_xy_sum = mul_xy_sum + float(x[i]) * float(y[i])
        x_sum = x_sum + float(x[i])
        y_sum = y_sum + float(y[i])
        x_sum_pow = x_sum_pow + pow(float(x[i]), 2)
        y_sum_pow = y_sum_pow + pow(float(y[i]), 2)
    try:
        r = ((n * mul_xy_sum) - (x_sum * y_sum))
              / math.sqrt( ((n*x_sum_pow) - pow(x_sum, 2)) * ((n*y_sum_pow) - pow(y_sum, 2)))
    except :
        print("error")
        r = 0.0
    return r
```

상관관계 분석 실습2

```
def main():
    h = [100, 200, 300, 400, 500, 600, 700, 800, 900]
    t = [18.0, 17.5, 17, 16.5, 15, 14.5, 13, 12, 11]
    print(h)
    print(t)
    print('상관분석:',correlation(h, t))
```

```
if __name__ == "__main__":
    main()
```

공공데이터를 이용한 상관분석

- 저장된 JSON 파일
  - 서울특별시_관광지입장정보_2011_2016.json
  - 중국(112)_해외방문객정보_2011_2016.json
  - 일본(130)_해외방문객정보_2011_2016.json
  - 미국(275)_해외방문객정보_2011_2016.json

공공데이터를 이용한 상관분석

```
import json
import math
import numpy as np

import matplotlib.pyplot as plt
import matplotlib
from matplotlib import font_manager, rc

import pandas as pd
```

```
#[CODE 1]
def correlation(x, y):
    n = len(x)
    vals = range(n)

    x_sum = 0.0
    y_sum = 0.0
    x_sum_pow = 0.0
    y_sum_pow = 0.0
    mul_xy_sum = 0.0

    for i in vals:
        mul_xy_sum = mul_xy_sum + float(x[i]) * float(y[i])
        x_sum = x_sum + float(x[i])
        y_sum = y_sum + float(y[i])
        x_sum_pow = x_sum_pow + pow(float(x[i]), 2)
        y_sum_pow = y_sum_pow + pow(float(y[i]), 2)

    try:
        r = ((n * mul_xy_sum) - (x_sum * y_sum)) / math.sqrt(
            ((n*x_sum_pow) - pow(x_sum, 2)) * ((n*y_sum_pow) - pow(y_sum, 2)) )
    except:
        r = 0.0
    return r
```

```
#[CODE 2]
def setScatterGraph(tour_table, fv_table, tourpoint):

    #[CODE 8]
    tour = tour_table[tour_table['resNm'] == tourpoint]
    merge_table = pd.merge(tour, fv_table, left_index=True, right_index=True)

    fig = plt.figure()

    fig.suptitle(tourpoint + '상관관계 분석')

    plt.subplot(1, 3, 1)
    plt.xlabel('중국인 입국수')
    plt.ylabel('외국인 입장객수')
    r1 = correlation(list(merge_table['china']), list(merge_table['ForNum']))
    plt.title('r = {:.5f}'.format(r1))
    plt.scatter(list(merge_table['china']), list(merge_table['ForNum']),
            edgecolor='none', alpha=0.75, s=6, c='black')
```

```
plt.subplot(1, 3, 2)
plt.xlabel('일본인 입국수')
plt.ylabel('외국인 입장객수')
r2 = correlation(list(merge_table['japan']), list(merge_table['ForNum']))
plt.title('r = {:.5f}'.format(r2))
plt.scatter(list(merge_table['japan']), list(merge_table['ForNum']),
        edgecolor='none', alpha=0.75, s=6, c='black')

plt.subplot(1, 3, 3)
plt.xlabel('미국인 입국수')
plt.ylabel('외국인 입장객수')
r3 = correlation(list(merge_table['usa']), list(merge_table['ForNum']))
plt.title('r = {:.5f}'.format(r3))
plt.scatter(list(merge_table['usa']), list(merge_table['ForNum']),
        edgecolor='none', alpha=0.75, s=6, c='black')

plt.tight_layout()

#fig = matplotlib.pyplot.gcf()
#fig.set_size_inches(10, 7)
#fig.savefig(tourpoint+'.png', dpi=300)

plt.show()
return [tourpoint, r1, r2, r3]
```

```
def main():

    font_location = "c:/Windows/fonts/malgun.ttf"
    font_name = font_manager.FontProperties(fname=font_location).get_name()
    matplotlib.rc('font', family=font_name)

    #[CODE 4]
    tpFileName = 'd:/data/data_al/서울특별시_관광지입장정보_2011_2016.json'
    jsonTP = json.loads(open(tpFileName, 'r', encoding='utf-8').read())
    tour_table = pd.DataFrame(jsonTP, columns=('yyyymm', 'resNm', 'ForNum'))
    tour_table = tour_table.set_index('yyyymm')

    #[CODE 5]
    resNm = tour_table.resNm.unique()
```

```
#[CODE 6]
fv_CFileName = 'd:/data/data_al/중국(112)_해외방문객정보_2011_2016.json'
jsonFV = json.loads(open(fv_CFileName, 'r', encoding='utf-8').read())
china_table = pd.DataFrame(jsonFV, columns=('yyyymm', 'visit_cnt'))
china_table = china_table.rename(columns={'visit_cnt': 'china'})
china_table = china_table.set_index('yyyymm')

fv_JFileName = 'd:/data/data_al/일본(130)_해외방문객정보_2011_2016.json'
jsonFV = json.loads(open(fv_JFileName, 'r', encoding='utf-8').read())
japan_table = pd.DataFrame(jsonFV, columns=('yyyymm', 'visit_cnt'))
japan_table = japan_table.rename(columns={'visit_cnt': 'japan'})
japan_table = japan_table.set_index('yyyymm')

fv_UFileName = 'd:/data/data_al/미국(275)_해외방문객정보_2011_2016.json'
jsonFV = json.loads(open(fv_UFileName, 'r', encoding='utf-8').read())
usa_table = pd.DataFrame(jsonFV, columns=('yyyymm', 'visit_cnt'))
usa_table = usa_table.rename(columns={"visit_cnt": "usa"})
usa_table = usa_table.set_index('yyyymm')
```

```
    #[CODE 7]
    fv_table = pd.merge(china_table, japan_table, left_index=True, right_index=True)
    fv_table = pd.merge(fv_table, usa_table, left_index=True, right_index=True)

    r_list = []
    for tourpoint in resNm:
        #[CODE 9]
        r_list.append(setScatterGraph(tour_table, fv_table, tourpoint))

    #[CODE 10]
    r_table = pd.DataFrame(r_list, columns=('tourpoint', 'china', 'japan', 'usa'))
    r_table = r_table.set_index('tourpoint')
    r_table = r_table.drop('서울시립미술관 본관')
    r_table = r_table.drop('서대문자연사박물관')
    r_table.plot(kind='bar', rot=70)
    plt.show()

if __name__ == "__main__":

    main()
```

5. 지도 시각화-Folium

- 포리움(Folium)
  - 'Open Street Map'과 같은 지도데이터에 'Leaflet.js'를 이용하여 위치정보를 시각화하기 위한 라이브러리
  - 기본적으로 'GeoJSON(http://geojson.org/)' 형식 또는 'topoJSON' 형식으로 데이터를 지정하면, 오버레이를 통해 마커의 형태로 위치 정보를 지도상에 표현할 수 있다.
  - 현실세계에서 지도상의 위치를 표시하기 위해 '위도(latitude)'와 '경도(longitude)'를 사용
  - 위도 : 적도를 기준으로 남쪽으로 남극점까지 90°, 북쪽으로 북극점까지 90°로 나누어 표시(우리나라 적도의 북쪽인 북위 34° ~ 38° 사이에 위치)
  - 경도: 런던 그리니치천문대를 지나는 본초 자오선을 중심으로 동서로 나누어 동경 180°, 서경 180°로 분리(서울의 경우 동경 127°에 위치).
  - folium 공식 깃허브 : https://github.com/python-visualization/folium
  - folium 공식 documentation : https://python-visualization.github.io/folium/

5. 지도 시각화-Folium

- GeoJSON'
  - 다양한 지리 데이터 구조를 인코딩하기 위한 형식을 제공
  - 지오메트리, 지형지물을 표시할 수 있으며, Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, GeometryCollection과 같은 속성들을 지정할 수 있음.
  - GeoJSON 형식

```
{
        "type": "Feature",
        "geometry": {
                "type": "Point",
                "coordinates": [125.6, 10.1]
        },
        "properties": {
                "name": "Dinagat Islands"
        }
}
```

5. 지도 시각화-Folium

- Folium 설치 및 버전확인

```
import os        #pip install folium
import numpy as np
import folium
from folium import plugins
print(folium.__version__)
```

- Folium 객체 생성

```
m = folium.Map(location=[37.566345, 126.977893])
m.save('data_al/map1.html') #파일이 저장될 위치
m
```

- zoo_start 속성 및 ScrollZoomToggler

```
m = folium.Map([37.566345, 126.977893], zoom_start=4)
plugins.ScrollZoomToggler().add_to(m)
m.save(os.path.join('results', 'Plugins_0.html'))
m
```

5. 지도 시각화-Folium

맵의 유형

- 포리움은 기본적으로 'Open Street Map'을 기반으로 동작
- 'Stamen Terrain', 'Stamen Toner', 'Mapbox Bright', 와 'Mapbox Control room tiles' 형식을 내장

```
map_osm = folium.Map(location=[37.566345, 126.977893], zoom_start=17,
tiles='Stamen Terrain')
map_osm.save('data_al/map3.html')
map_osm = folium.Map(location=[37.566345, 126.977893], zoom_start=17,
tiles='Stamen Toner')
map_osm.save('data_al/map4.html')
```

- Cloudmade'나 'Mapbox'를 사용하는 경우에는 사이트에 등록시 발급받은 API 키 정보를 아래와 같이 'API_key' 속성으로 지정

```
map_osm = folium.Map(location=[37.5660, 126.9757], tiles='Mapbox',
API_key='API키값')
```

5. 지도 시각화-Folium

- 마커(Marker)와 팝업(Popup)의 설정
  - 포리움은 다양한 형식의 마커(특정 위치를 표시하는 표식)과 마커를 클릭하였을 때 나타나는 정보(Popup)을 지정할 수 있다.
  - Marker()’ 메소드를 이용하여 생성
  - 마커의 인자값으로 위경도 값 리스트와 마커를 클릭할 시 보여줄 문자열을 전달하고, 생성한 포리움 객체에 추가(.add_to())하면 간단하게 마커를 생성

```
map_osm = folium.Map(location=[37.566345, 126.977893], zoom_start=17)
folium.Marker([37.566345, 126.977893], popup='서울특별시청')
            .add_to(map_osm)
folium.Marker([37.5658859, 126.9754788], popup='덕수궁').add_to(map_osm)
map_osm.save('data_al/map5.html')
```

5. 지도 시각화-Folium

- 마커(Marker)와 팝업(Popup)의 설정
  - 포리움 마커는 부트스트랩(bootstrap)을 이용, 아이콘 타입을 설정할 수 있으며, 범위를 설정하기 위하여 circle 속성을 줄 수 있다.
  - 덕수궁의 위치를 좀더 크게 마커로 표시하고, 서울특별시청은 적색의 'info-sign' 마커로 표시한 예

```
m=folium.Map(location=[37.566345, 126.977893], zoom_start=17)
folium.Marker([37.566345, 126.977893], popup='서울특별시청',
icon=folium.Icon(color='red',icon='info-sign')).add_to(m)
folium.CircleMarker([37.5658859, 126.9754788], radius=100,
color='#3186cc', fill_color='#3186cc', popup='덕수궁').add_to(m)
map_osm.save('data_al/map6.html')
```

마커(Marker)와 팝업(Popup)의 설정

```
N = 100
data = np.array(
    [
        np.random.uniform(low=35, high=38, size=N), # Random latitudes.
        np.random.uniform(low=125, high=129, size=N), # Random longitudes.
    ] ).T
popups = [str(i) for i in range(N)] # Popups texts are simple numbers.
m = folium.Map([37.566345, 126.977893], zoom_start=8)
plugins.MarkerCluster(data, popups=popups).add_to(m)
m.save(os.path.join('results', 'Plugins_1.html'))
m
```

- Terminator

```
m = folium.Map([37.566345, 126.977893], zoom_start=1)
plugins.Terminator().add_to(m)
m.save(os.path.join('results', 'Plugins_2.html'))
m
```

- BoatMarker

```
m = folium.Map([37.566345, 126.977893], zoom_start=17)
plugins.BoatMarker(location=(37.566345, 126.977893), heading=45,
        wind_heading=150, wind_speed=45, color='#8f8' ).add_to(m)
plugins.BoatMarker(location=(37.5658859, 126.9754788), heading=-20,
        wind_heading=46, wind_speed=25, color='#88f' ).add_to(m)
m.save(os.path.join('results', 'Plugins_3.html'))
m
```

- BeautifyIcon

```
m = folium.Map([45.5, -122], zoom_start=3)
icon_plane = plugins.BeautifyIcon(
    icon='plane',
    border_color='#b3334f',
    text_color='#b3334f',
    icon_shape='triangle')
icon_number = plugins.BeautifyIcon(
    border_color='#00ABDC',
    text_color='#00ABDC',
    number=10,
    inner_icon_style='margin-top:0;')
folium.Marker(
    location=[46, -122],
    popup='Portland, OR',
    icon=icon_plane ).add_to(m)
folium.Marker(
    location=[50, -122],
    popup='Portland,OR', icon=icon_number ).add_to(m)
m.save(os.path.join('results', 'Plugins_4.html'))
m
```

- Fullscreen

```
m = folium.Map(location=[41.9, -97.3], zoom_start=4)
plugins.Fullscreen(
    position='topright',
    title='Expand me',
    title_cancel='Exit me',
    force_separate_button=True ).add_to(m)
m.save(os.path.join('results', 'Plugins_5.html'))
m
```

5. 지도 시각화-Folium

- JSON 활용 데이터
  - geoJSON 형식과 topoJSON은 지도상의 경계 영역 등을 표시하기에 효율적
  - 행정구역이 “MultiPolygon“ 형식으로 입력되어 있는 JSON 데이터를 불러들여 지도에 표시한 예

```
import folium
import json
m = folium.Map(location=[37.566345, 126.977893])
rfile = open('data_al/skorea_provinces_geo_simple.json','r', encoding='utf-8').read()
jsonData = json.loads(rfile)
folium.GeoJson(jsonData, name='json_data').add_to(m)
m.save('d:/data/data_al/map7.html')
m
```