

MINI PROJECT

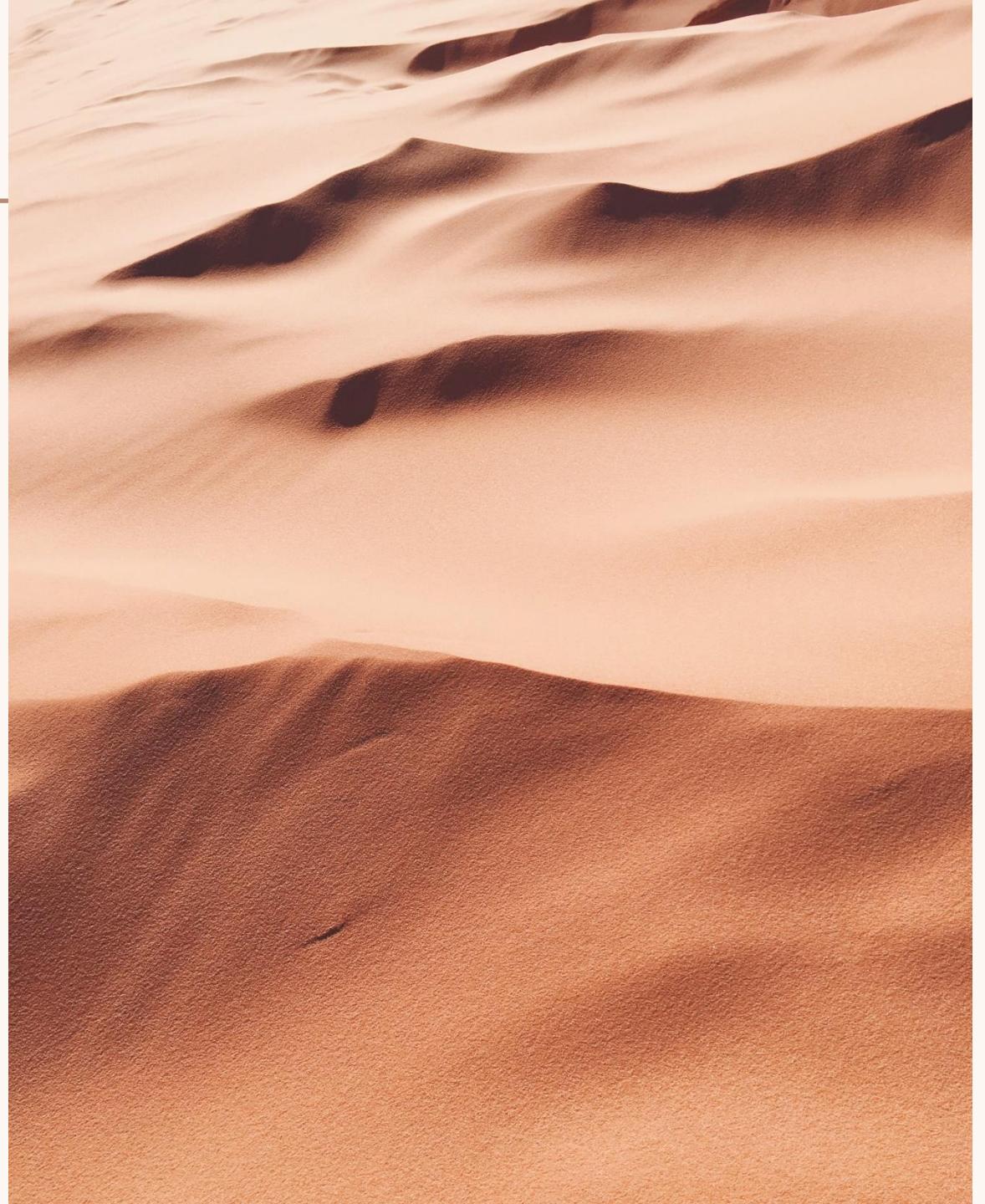
| Crawling & Visualization

Table of Contents

1 Topic

2 Crawling and
Data processing

3 Visualization



Part 1.

Topic



Topic_시장 현황

“신종 코로나바이러스 감염증(코로나19)으로 전년 동기 대비 국내 숙소 사전 예약 건수 109% 증가”

“수도권에서 자가용으로 이동가능한 강원도(16.9%)와 경기도(14.9%)를 비롯한
여름철 인기 지역인 제주도(14.3%)와 부산(9.8%)이 상위권을 차지”

“펜션(43.8%)이 5월 황금연휴 기간에 이어 또 한 번 이용률 1위를 기록.
타인과의 접촉 가능성이 낮은 독채형 숙소에 대한 선호도가 지속”

(출처 : 야놀자, 2020)

“코로나 변이 바이러스 등장으로 연말 국내 호텔 투숙 수요 지난해 대비 3배 이상 급증”

“21년 연말 연휴 기간 결제 금액 지난해 대비 호텔 25.6%, 펜션 18.3%, 캠핑 23.6% 상승”

“21년 연말 호텔 예약 건수 3.2배 폭증, 입실일 기준 평균 28.4일 전 예약 완료”

(출처 : 여기어때, 2021)

Topic_선정 이유

“2021 Best Pension site in South Korea”



코로나 장기화로 많은 사람들은 국내 여행지를 찾고 있다.



다양한 여행 어플과 웹 사이트에서 숙박 예약 서비스를 제공하고 있지만,
많은 정보량으로 원하는 숙소를 찾기가 쉽지 않다.



그 중에서도 펜션은 SNS(네이버, 인스타그램 등) 리뷰를
일일이 찾아봐야 하는 번거로움이 있다.

Topic_프로젝트 목표



1) 네이버 여행 인플루언서의
블로그 리뷰를 크롤링하여
전국의 펜션 리뷰 데이터를 확보한다

2) 데이터 전처리 과정으로
시각화를 위한 데이터로 가공한다

3) 다양한 시각화 패키지를 활용해
인사이트를 발견하고
최적의 정보전달 방법을 찾는다

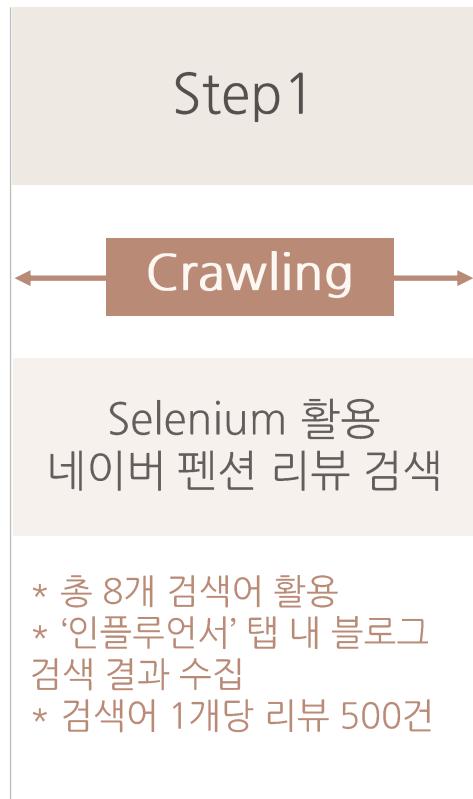
Part 2:

Crawling and
Data processing

Crawling and Data processing_전체 과정

Step1	Step2	Step3	Step4	Step5
Selenium 활용 네이버 펜션 리뷰 검색	전체 리뷰 URL 및 기타 정보 수집	각 블로그 URL 내 리뷰 및 펜션정보 수집	결측치 확인 및 대체 새로운 컬럼 추가	시각화를 위한 데이터프레임 추가 생성
<ul style="list-style-type: none"> * 총 8개 검색어 활용 * ‘인플루언서’ 탭 내 블로그 검색 결과 수집 * 검색어 1개당 리뷰 500건 	<ul style="list-style-type: none"> * 리뷰 URL, 업로드 날짜 * 블로거 이름, 팬 수 	<ul style="list-style-type: none"> * 리뷰 제목, 본문 내용 * 네이버 맵 위젯 내의 펜션 이름, 전화번호, 주소, 위치정보, Place_ids 수집 	<ul style="list-style-type: none"> * place_ids와 place_tels 컬럼만 결측치 존재하여 0으로 대체 * 년도, 월, 일, 팬 level, 시도 컬럼 추가 	<ul style="list-style-type: none"> * 2021년 데이터만 활용 * 월, 시도, 계절, 펜션타입 총 4가지 컬럼으로 구성된 게시글 수 데이터프레임 생성

Crawling and Data processing_세부 과정



N 풀빌라 펜션

통합 VIEW 이미지 지식iN **인플루언서** 동영상 쇼핑 뉴스 어학사전 지도 ...

인플루언서 참여 콘텐츠

 **소치미** 여행 전문블로거 | 국내 전문 | 팬 1.1만+
b 블로그 이웃 2.9만+ · SNS 핫플 매니아 · 여행 리뷰



대부도 풀빌라 여기바다 온수 수영장, 노래방 인천 독채 펜션 ❤️
온수 수영장, 노래방, 바베큐까지 있는 곳, 3층을 통째로 쓰는 인천 펜션 체크인 14:00 체크아웃 11:00 최대 6인, 유아 동반 가능 한 쪽 벽면이 모두 통유리로 되어 있어서 묵는 내내 바다를 원없이 볼 수 있었던 대부도...
4일 전

b 포항 풀빌라 펜션 하루 / 오션뷰 수영장, 스파 신축 숙소 ❤️
b 공주 숙소 스테이오안 / 신축 온수풀 독채 펜션 풀빌라 감성 최고 ❤️

소치미의 참여 콘텐츠 2개 더보기 >

Crawling and Data processing_세부 과정

Step1

Crawling

Selenium 활용
네이버 펜션 리뷰 검색

- * 총 8개 검색어 활용
- * ‘인플루언서’ 탭 내 블로그
검색 결과 수집
- * 검색어 1개당 리뷰 500건

① 네이버 열고 검색어 입력

```
search_word = input("검색어를 입력하세요:")
driver = webdriver.Chrome('./chromedriver')
driver.get("https://www.naver.com/")
driver.implicitly_wait(1)

elem = driver.find_element_by_class_name("input_text")
elem.send_keys(search_word)
elem.send_keys(Keys.RETURN)

driver.implicitly_wait(2)
```

검색어를 입력하세요:풀빌라 펜션

③ 스크롤 가장 하단으로 내리기

```
last_height = driver.execute_script("return document.documentElement.scrollHeight")

while True:

    driver.execute_script("window.scrollTo(0, document.documentElement.scrollHeight);")
    time.sleep(2)

    new_height = driver.execute_script("return document.documentElement.scrollHeight")
    if new_height == last_height:    #last_height에 원하는 수치 입력
        break
    last_height = new_height

driver.execute_script("window.scrollTo(0, document.documentElement.scrollHeight);")
```

② 인플루언서 탭 클릭

```
a = driver.find_element(By.CSS_SELECTOR, "li.menu:nth-child(5) > a")
b = driver.find_element(By.CSS_SELECTOR, "li.menu:nth-child(7) > a")

print(a.text)

if a.text == "인플루언서NEW":
    a.click()
else:
    b.click()

time.sleep(1)

인플루언서NEW
```

Crawling and Data processing_세부 과정



① 문자열을 날짜 형식으로 변환하는 함수 정의

```
## 문자열을 날짜로 변환하는 함수

def str2date(date_str):

    if date_str[1:] == "일 전":      #일주일 이내 등록된 게시글
        date = datetime.today() - timedelta(days=int(date_str[0]))  #업로드 날짜 계산
        return date

    elif (date_str == "어제") or (date_str[2:] == "시간 전"):
        date = datetime.today() - timedelta(days=1)  #업로드 날짜 계산
        return date

    elif (date_str[1:] == "시간 전") or (date_str[1:] == "분전") or (date_str[2:] == "분전"):
        date = datetime.today()
        return date

    else:
        date_str = date_str.replace(".", "-")[:-1]  #"%Y-%m-%d" 형식으로 변환
        date = datetime.strptime(date_str,"%Y-%m-%d")  #문자열을 날짜 형식으로 변환
        return date

## 함수 돌아가면 datetime 형식 날짜 리턴
```

Crawling and Data processing_세부 과정

② 전체 리뷰 URL 및 업로드 날짜, 블로거 이름, 팬 수 정보 수집

Step2

Crawling

전체 리뷰 URL 및
기타 정보 수집

- * 리뷰 URL, 업로드 날짜
- * 블로거 이름, 팬 수

```
### 크롤링 코드 ###

dates = []
blooger_names = []
fan_counts = []
urls = []

## 업로드 날짜
span_dates = driver.find_elements(By.CSS_SELECTOR, "div.detail_box._cross_trigger span.date")

for each in span_dates:
    date_str = each.text      #텍스트만 가져오기
    result_date = str2date(date_str)  #문자열인 date를 넣으면 datetime 형식으로 변환하는 함수 사용
    dates.append(result_date)

## 블로거 이름
names = driver.find_elements(By.CSS_SELECTOR, "div.user_box_inner a.name.ellipsis")

for each in names:
    name = each.text
    blooger_names.append(name)

## 팬 수
fans = driver.find_elements(By.CSS_SELECTOR, "div.user_box_inner span._fan_count")

for each in fans:
    fan = each.text
    fan_counts.append(fan)

## 블로그 url
a_links = driver.find_elements(By.CSS_SELECTOR, "div.dsc_area > a.name_link")

for each in a_links:
    url = each.get_attribute("href")
    urls.append(url)
```

Crawling and Data processing_세부 과정

③ 필요 없는 데이터 삭제 및 팬 수 정수로 변환

Step2

Crawling

전체 리뷰 URL 및
기타 정보 수집

* 리뷰 URL, 업로드 날짜
* 블로거 이름, 팬 수

```
## 500개 초과된 변수가 존재한다면
## 앞쪽에서부터 500개로 맞춰서 슬라이싱 후 데이터프레임 생성

if len(bloger_names) > 500:
    bloger_names = bloger_names[len(bloger_names)-500:]

if len(fan_counts) > 500:
    fan_counts = fan_counts[len(fan_counts)-500:]

if len(urls) > 500:
    urls = urls[len(urls)-500:]

fans = []
for i in range(len(fan_counts)):
    count = re.sub("[만]?", "", "000", fan_counts[i])
    count = int(re.sub("[.,]", "", count))
    fans.append(count)
```

④ 데이터 프레임으로 저장 'search_blogs'

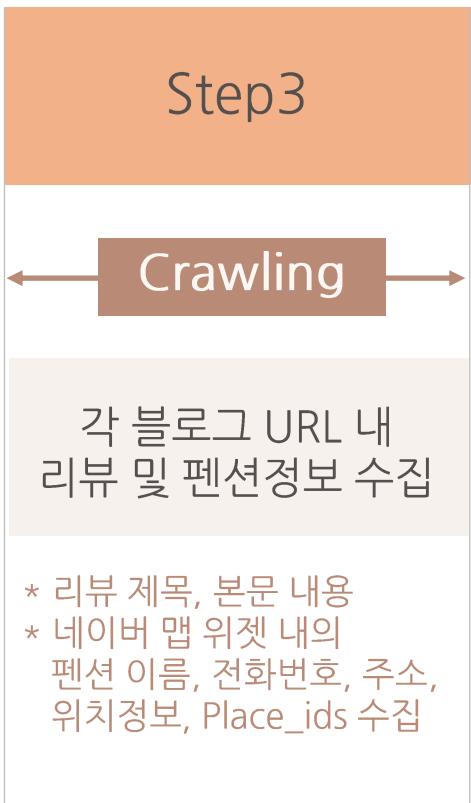
```
search_blogs = pd.DataFrame()

search_blogs['dates'] = dates
search_blogs['bloger_names'] = bloger_names
search_blogs['fans'] = fans
search_blogs['urls'] = urls

search_blogs.head()
```

	dates	bloger_names	fans	urls
0	2021-12-27	열씨미	14000	https://in.naver.com/jbm993/contents/internal/...
1	2021-12-24	망고요정	13000	https://in.naver.com/mango/contents/internal/3...
2	2021-12-23	유에	1000	https://in.naver.com/rafiuta/contents/internal...
3	2021-12-02	일상탈출	13000	https://in.naver.com/tcacyc/contents/internal/...
4	2021-11-21	예릭샘	11000	https://in.naver.com/usgreen99/contents/internal...

Crawling and Data processing_세부 과정



①-1 블로그 URL 하나씩 열어서 블로그인지(조건1), 네이버 맵 위젯이 존재하는지(조건2)
확인해서 리뷰 제목 및 본문 가져오기 (이어서)

```

titles = []
main_contents = []
place_ids = []
place_names = []
place_addrs = []
place_tels = []
lats = []
lons = []
urls = []
blog_urls = []

```

```

### 크롤링 코드 ####
for url in range(len(search_blogs)):

    ## 블로그 url 열기
    driver = webdriver.Chrome('.\chromedriver', chrome_options = options)
    driver.get(search_blogs['urls'][url])
    time.sleep(1)

    ## 블로그인지 url 받아와서 확인
    current_url = driver.current_url

    if current_url[:23] == "https://blog.naver.com/":

        ## 본문 내용이 있는 iframe(mainFrame)으로 들어가기
        driver.switch_to.frame("mainFrame")

        ## map_info 있는지 확인
        check_elements = driver.find_elements(By.CSS_SELECTOR, "div.se-module > a.se-map-info")

        ## map_info 있으면 블로그 글 제목, 본문 내용, 펜션 이름, 위도, 경도 데이터 저장
        if len(check_elements) != 0:

            ## 네이버 맵 정보 가져오기
            map_info = driver.find_element(By.CSS_SELECTOR, "div.se-module > a.se-map-info")
            map_data = map_info.get_attribute("data-linkdata")
            map_dict = json.loads(map_data) # str 형태의 map_data를 dict 형태로 변환
            time.sleep(1)

            ## 블로그 글 제목 가져오기
            title = driver.find_element(By.CSS_SELECTOR, "div.pc01").text
            titles.append(title)
            print(title)

            ## 블로그 본문 내용 가져오기
            contents = driver.find_elements(By.CSS_SELECTOR, "div.se-component.se-text.se-l-default")
            texts = []
            for each in contents:
                texts.append(each.text) #contents 안에 있는 text만 가져와서 저장
            text = " ".join(texts) #list 요소로 저장된 문장을 하나로 이어붙이기
            main_contents.append(text)

```

Crawling and Data processing_세부 과정

①-2 (이어서) 네이버 맵 위젯 내의 Place_ids, 펜션 이름, 주소, 전화번호, 위도, 경도, URL 저장하기

Step3

Crawling

각 블로그 URL 내
리뷰 및 펜션정보 수집

- * 리뷰 제목, 본문 내용
- * 네이버 맵 위젯 내의 펜션 이름, 전화번호, 주소, 위치정보, Place_ids 수집

```
## 펜션 정보 가져오기
place_ids.append(map_dict['placeId'])
place_names.append(map_dict['name'])
place_addrs.append(map_dict['address'])
place_tels.append(map_dict['tel'])
lats.append(map_dict['latitude'])
lons.append(map_dict['longitude'])
```

```
## url 저장하기
url = search_blogs['urls'][url]
urls.append(url)
current_url = driver.current_url
blog_urls.append(current_url)
```

```
## 웹 드라이버 종료
driver.close()
```

```
else:
    driver.close()
else:
    driver.close()
```

경남 펜션 오션뷰 숙소 사천 풀빌라 독채 펜션
 부산 펜션 송정해수욕장 풀빌라 호텔 후기
 남해 숙소 추천 강성 오션뷰 수영장 노천탕 남해 풀빌라 펜션 웨이포인트풀빌라
 경남 펜션 강성 넘치는 사천 풀빌라 독채펜션
 대부도 풀빌라 디디모스 키즈 펜션 리얼 후기
 여수 독채 풀빌라 펜션 온수풀 수영장
 국내 강성숙소 충북 제천 풀빌라 펜션 수페23 숙박 후기
 충청도 신상 숙소 독채 풀빌라 모음 : 아산 스테이느느루, 청주 휘계, 공주 스테이,
 영덕 펜션 하저스테이 :: 포항 근교 독채 풀빌라 가족여행
 포항 풀빌라 펜션 하루 / 오션뷰 수영장, 스파 신축 숙소 ♥
 남해 신축 강성숙소 디풀빌라 펜션, 돌남아 느낌 낭낭 🌳

Crawling and Data processing_세부 과정

Step3

Crawling

각 블로그 URL 내
리뷰 및 펜션정보 수집

- * 리뷰 제목, 본문 내용
- * 네이버 맵 위젯 내의
펜션 이름, 전화번호, 주소,
위치정보, Place_ids 수집

② 데이터프레임으로 저장 'blog_df'

```
blog_df = pd.DataFrame()
blog_df['titles'] = titles
blog_df['main_contents'] = main_contents
blog_df['place_ids'] = place_ids
blog_df['place_names'] = place_names
blog_df['place_addrs'] = place_addrs
blog_df['place_tels'] = place_tels
blog_df['lats'] = lats
blog_df['lons'] = lons
blog_df['urls'] = urls
blog_df['blog_urls'] = blog_urls
```

③ 'search_blogs' 데이터 프레임과 결합 한 후 최종 데이터프레임으로 저장 (=검색어 1개 완료)

```
: result_df = pd.merge(blog_df, search_blogs, on='urls', how='left')
result_df.head()
```

	titles	main_contents	place_ids	place_names	place_addrs	place_tels	lats	lons	urls	blog_urls	
0	경남 펜션 오션 뷰 숙 소 사 천 빌 독 펜	경남 펜션 오션 뷰 숙소 사천 뷰 빌라 독채 펜션 펜캉스 힐링 경 남 독채펜션 ~\n11월...		사천리조트 뷰 빌라펜션		경상남도 사 천시 토끼로 245 비토섬 신우리조트	None	34.9844337	127.9663409	https://in.naver.com/jbm993/contents/internal/...	https://blog...
1	부산 펜션 송정 해수 욕장 풀 라 후 기	Busan - 온수풀 풀빌라 있는데 가성비 좋은 속 소 - 풀 해는 곳 곳에 화이트 크 리...	1102393804	그레이션즈 뷰 빌라 펜션	부산광역시 해운대구 송 정해변로 30- 1 그레이션즈 2층 3층 4층 5층	010-4074- 0790	35.1809217	129.202743	https://in.naver.com/mango/contents/internal/3...	https://blog...	

Crawling and Data processing_세부 과정

① 모든 검색결과 (8개 검색어) 불러와서 type 컬럼 추가 후
전체 결합한 데이터프레임 생성 'pension_df'

Step4

← Data processing →

결측치 확인 및 대체
새로운 컬럼 추가

- * place_ids와 place_tels
컬럼만 결측치 존재하여
0으로 대체
- * 년도, 월, 일, 팬 level,
시도 컬럼 추가

```
pension_couple = pd.read_csv("pension_couple.csv", encoding='utf-8')
pension_dog = pd.read_csv("pension_dog.csv", encoding='utf-8')
pension_family = pd.read_csv("pension_family.csv", encoding='utf-8')
pension_kids = pd.read_csv("pension_kids.csv", encoding='utf-8')
pension_nokids = pd.read_csv("pension_nokids.csv", encoding='utf-8')
pension_ocean = pd.read_csv("pension_ocean.csv", encoding='utf-8')
pension_pool = pd.read_csv("pension_pool.csv", encoding='utf-8')
pension_spa = pd.read_csv("pension_spa.csv", encoding='utf-8')
```

```
pension_couple['type'] = 'cpl'
pension_dog['type'] = 'dog'
pension_family['type'] = 'fmy'
pension_kids['type'] = 'kids'
pension_nokids['type'] = 'nokids'
pension_ocean['type'] = 'ocean'
pension_pool['type'] = 'pool'
pension_spa['type'] = 'spa'
```

```
pension_list = [pension_couple, pension_dog, pension_family, pension_kids, pension_nokids, pension_ocean, pension_pool, pension_spa]
pension_df = pd.concat(pension_list, ignore_index=True)

pension_df.shape
```

Crawling and Data processing_세부 과정

Step4

← Data processing →

결측치 확인 및 대체 새로운 컬럼 추가

- * place_ids와 place_tels
컬럼만 결측치 존재하여
0으로 대체
- * 연도, 월, 일, 팬 level,
시도 컬럼 추가

② 연도, 월, 일 및 계절 컬럼 추가

```
pension_df['year'] = pension_df['dates'].dt.year
pension_df['month'] = pension_df['dates'].dt.month
pension_df['day'] = pension_df['dates'].dt.day
```

```
pension_df['season'] = 'season'

pension_df['season'][pension_df['month']==12] = 'win' #겨울
pension_df['season'][pension_df['month']==1] = 'win'
pension_df['season'][pension_df['month']==2] = 'win'
pension_df['season'][pension_df['month']==3] = 'spr' #봄
pension_df['season'][pension_df['month']==4] = 'spr'
pension_df['season'][pension_df['month']==5] = 'spr'
pension_df['season'][pension_df['month']==6] = 'sum' #여름
pension_df['season'][pension_df['month']==7] = 'sum'
pension_df['season'][pension_df['month']==8] = 'sum'
pension_df['season'][pension_df['month']==9] = 'aut' #가을
pension_df['season'][pension_df['month']==10] = 'aut'
pension_df['season'][pension_df['month']==11] = 'aut'

pension_df['season'].head()
```

```
0    win
1    aut
2    aut
3    win
4    win
Name: season, dtype: object
```

③-1 결측치 확인(place_ids, place_tels만 결측치 존재)

pension_df.info() #place_ids, place_tels만 결측치 존재

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2833 entries, 0 to 2832
Data columns (total 18 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   titles      2833 non-null   object 
 1   main_contents 2833 non-null   object 
 2   place_ids    2791 non-null   float64
 3   place_names  2833 non-null   object 
 4   place_addrs  2833 non-null   object 
 5   place_tels   2664 non-null   object 
 6   lats         2833 non-null   float64
 7   longs        2833 non-null   float64
 8   urls         2833 non-null   object 
 9   blog_urls   2833 non-null   object 
 10  dates        2833 non-null   object 
 11  bloger_names 2833 non-null   object 
 12  fans         2833 non-null   int64  
 13  year         2833 non-null   int64  
 14  month        2833 non-null   int64  
 15  day          2833 non-null   int64  
 16  type         2833 non-null   object 
 17  season       2833 non-null   object 
dtypes: float64(3), int64(4), object(11)
memory usage: 398.5+ KB
```

Crawling and Data processing_세부 과정

Step4

← Data processing →

결측치 확인 및 대체 새로운 컬럼 추가

- * place_ids와 place_tels
컬럼만 결측치 존재하여
0으로 대체
- * 년도, 월, 일, 팬 level,
시도 컬럼 추가

③-2 결측치 모두 0으로 대체

```
pension_df = pension_df.fillna(0) #결측치 모두 0으로 대체
```

```
pension_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2833 entries, 0 to 2832
Data columns (total 18 columns):
 #   Column      Non-Null Count  Dtype  
 ---  --          --          --    
 0   titles       2833 non-null   object  
 1   main_contents 2833 non-null   object  
 2   place_ids     2833 non-null   float64 
 3   place_names   2833 non-null   object  
 4   place_addrs   2833 non-null   object  
 5   place_tels    2833 non-null   object  
 6   lats          2833 non-null   float64 
 7   lons          2833 non-null   float64 
 8   urls          2833 non-null   object  
 9   blog_urls    2833 non-null   object  
 10  dates         2833 non-null   object  
 11  bloger_names 2833 non-null   object  
 12  fans          2833 non-null   int64   
 13  year          2833 non-null   int64   
 14  month         2833 non-null   int64   
 15  day           2833 non-null   int64   
 16  type          2833 non-null   object  
 17  season        2833 non-null   object  
dtypes: float64(3), int64(4), object(11)
memory usage: 398.5+ KB
```

④ 2021년 데이터만 사용

```
## 2021년 데이터만 슬라이싱
```

```
pension_2021 = pension_df[pension_df['year'] == 2021]
pension_2021.head()
```

	titles	main_contents	place_ids	place_names	place_addrs	place_tels	lats
0	제주도 커 플펜션 추 천/ 제주N 스테이, 바 다가 보이 는 예쁜 공 간에서 하 루를...	제주도 커플펜 션 추천/ 제주N 스테이 언제라 도 갈대마다 마 을이 편안해지 고 즐거워지는 ...	1.467548e+09	제주N스테이	제주특별자 치도 제주시 구좌읍 한동 북1길 60 C 동	0507- 1324-0978	33.542778
1	제주 커플 펜션 오션 뷰 뷰스팟 제주도 서 쪽 숙소	제주 커플펜션 오션뷰 뷰스팟 제주도 서쪽 숲 소 제주도 서쪽 숙소 저바다에 누워~\n...	1.079993e+09	저바다에누워	제주특별자 치도 제주시 한경면 판조 로 3-5	0507- 1440-7904	33.367228
2	제주 오션 뷰 멋진 제 주도 커플 펜션 숙소	가을 제주가 너 무 아름다워 아 주 짙게 와이프 와 제주도를 다 시 다녀왔습니 다. 1박 2...	1.347007e+09	저바다에누워	제주특별자 치도 제주시 한경면 판조 로 3-5	0	33.367236

Crawling and Data processing_세부 과정

Step4

← Data processing →

결측치 확인 및 대체 새로운 컬럼 추가

- * place_ids와 place_tels
컬럼만 결측치 존재하여
0으로 대체
- * 년도, 월, 일, 팬 level,
시도 컬럼 추가

⑤ 팬 수를 4개 level로 구분해서 컬럼 추가

```
pension_2021['level'] = 0

for i in range(len(pension_2021)):

    if pension_2021['fans'][i] < 2500:
        pension_2021.loc[i,'level'] = 1
    elif pension_2021['fans'][i] < 5000:
        pension_2021.loc[i,'level'] = 2
    elif pension_2021['fans'][i] < 10000:
        pension_2021.loc[i,'level'] = 3
    else:
        pension_2021.loc[i,'level'] = 4

pension_2021['level'].head()
```

0	1
1	4
2	4
3	1
4	1

Name: level, dtype: int64

⑥ 펜션 주소에서 시도명만 추출해서 컬럼 추가

```
for i in range(len(pension_2021)):
    index = pension_2021.loc[i,'place_addrs'].find(" ")
    pension_2021.loc[i,'geo'] = pension_2021.loc[i,'place_addrs'][:index]

pension_2021.head()
```

	dates	bloger_names	fans	year	month	day	type	season	level	geo
	2021-12-30 16:46:59.995526	꼬미의 제주도 여행	1128	2021	12	30	cpl	win	1	제 주 특 별 자 치 도

2021-10-31 00:00:00.000000	열씨미	14000	2021	10	31	cpl	aut	4
-------------------------------	-----	-------	------	----	----	-----	-----	---

2021-10-27 00:00:00.000000	울트라오렌지	15000	2021	10	27	cpl	aut	4
-------------------------------	--------	-------	------	----	----	-----	-----	---

Crawling and Data processing_세부 과정

Step5

← Data processing →

시각화를 위한
데이터프레임 추가 생성

- * 2021년 데이터만 활용
- * 월, 시도, 계절, 펜션타입
총 4가지 컬럼으로 구성된
게시글 수 데이터프레임
생성

① 월, 시도, 계절, 펜션타입, 총 4가지 컬럼으로 구성된 게시글 수 데이터 프레임 생성

```
## 월, 타입, 계절, 시도별 게시글 수 그룹 df 생성
```

```
group = pension_2021.groupby(["month","type","season","geo"])
p_group = group.size().reset_index(name='post_count')
```

```
p_group
```

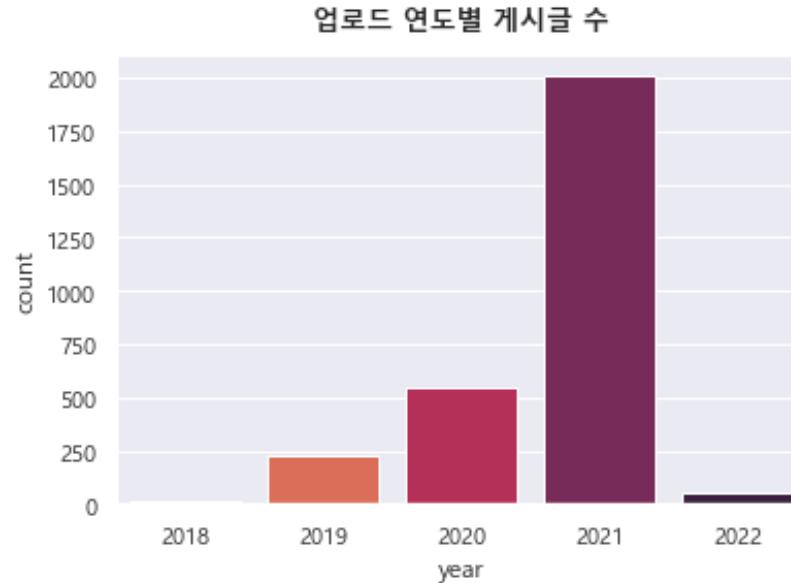
	month	type	season	geo	post_count
0	1	cpl	win	강원도	1
1	1	cpl	win	경상남도	1
2	1	cpl	win	경상북도	1
3	1	cpl	win	인천광역시	1
4	1	cpl	win	제주특별자치도	1
...
612	12	spa	win	경상남도	5
613	12	spa	win	경상북도	4
614	12	spa	win	인천광역시	3
615	12	spa	win	전라남도	1
616	12	spa	win	전라북도	1

617 rows × 5 columns

Part 3.

Visualization

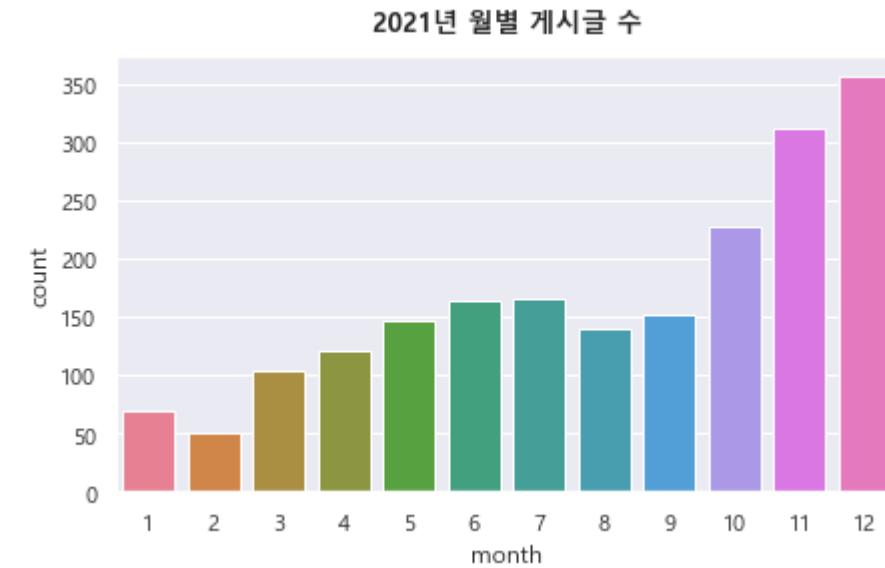
Visualization_Seaborn 활용



```
## 연도, 월별 게시글 수 #★
fig,ax = plt.subplots(figsize=(6,4))

sns.set(font_scale = 0.7)
sns.set_palette("rocket_r",n_colors=5, color_codes=True)
plt.rcParams['font.family'] = 'Malgun Gothic'

sns.countplot(x="year", data=pension_df,ax=ax);
plt.title("업로드 연도별 게시글 수", fontsize=13,
          fontweight='bold', y=1.03);
```

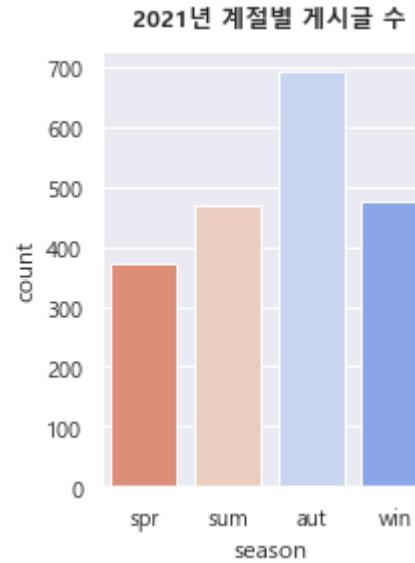


```
## 월별 게시글 수(2021년만) #★
fig,ax = plt.subplots(figsize=(7,4))

sns.set(font_scale = 1.0)
sns.set_palette("husl",n_colors=12, color_codes=True)
plt.rcParams['font.family'] = 'Malgun Gothic'

sns.countplot(x="month", data=pension_2021,ax=ax);
plt.title("2021년 월별 게시글 수", fontsize=13,
          fontweight='bold', y=1.03);
```

Visualization_Seaborn 활용



```
## 계절별 게시글 수(2021년만) #★
fig,ax = plt.subplots(figsize=(3,4))

sns.set(font_scale = 1.0)
sns.set_palette("coolwarm_r",n_colors=4, color_codes=True)
plt.rcParams['font.family'] = 'Malgun Gothic'

sns.countplot(x="season", data=pension_2021,ax=ax,
               order=['spr','sum','aut','win']);

plt.title("2021년 계절별 게시글 수", fontsize=12,
          fontweight='bold', y=1.03);
```

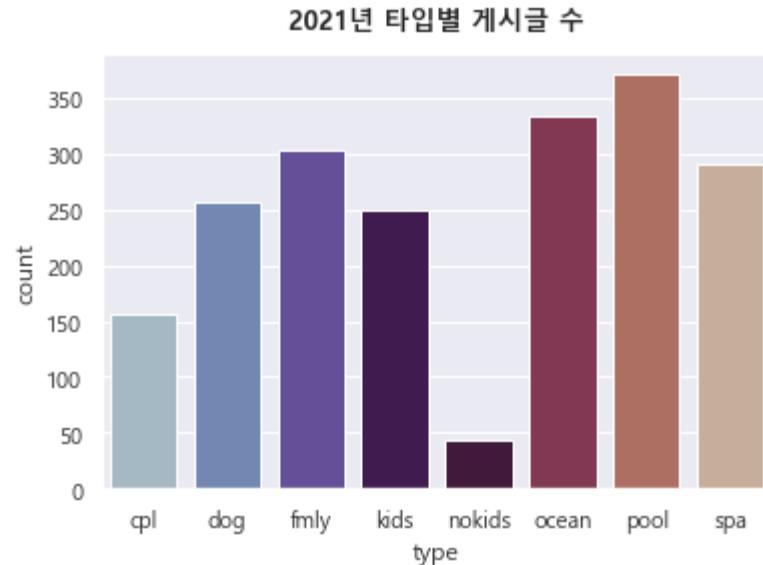
```
## 시도별 게시글 수(2021년만) #★
fig,ax = plt.subplots(figsize=(20,8))

sns.set(font_scale = 1.2)
sns.set_palette("tab20",n_colors=15, color_codes=True)

sns.countplot(x="geo", data=pension_2021,ax=ax);

plt.rcParams['font.family'] = 'Malgun Gothic'
plt.title("2021년 시도별 게시글 수", fontsize=20,
          fontweight='bold', y=1.03);
```

Visualization_Seaborn 활용



```
## 펜션 타입별 게시글 수(2021년만) ##★  
fig,ax = plt.subplots(figsize=(6,4))  
  
sns.set(font_scale = 1.0)  
sns.set_palette("twilight",n_colors=8, color_codes=True)  
plt.rcParams['font.family'] = 'Malgun Gothic'  
  
sns.countplot(x="type", data=pension_2021,ax=ax);  
plt.title("2021년 타입별 게시글 수", fontsize=13,  
          fontweight='bold', y=1.03);
```

같은 내용을 파이 차트로 만들면?

Visualization_Seaborn 활용

```

## 데이터 준비 #★
labels = pension_div['ctp_name'] ## 라벨
frequency = pension_div['post_count'] ## 빈도
colors = sns.color_palette('pastel')[0:len(labels)] ## 색상

## 파이차트 그리기
fig = plt.figure(figsize=(8,8)) ## 캔버스 생성
fig.set_facecolor('white') ## 캔버스 배경색을 하얀색으로 설정
ax = fig.add_subplot() ## 프레임 생성

pie = ax.pie(frequency, ## 파이차트 출력
              startangle=90, ## 시작점을 90도(degree)로 지정
              counter-clock=False, ## 시계방향으로 그려짐
              colors = colors, ## 색상 지정
              wedgeprops={'width': 0.7, 'edgecolor': 'w', 'linewidth': 5}) # 스타일 변경

total = np.sum(frequency) ## 빈도수 합
threshold = 5 ## 상한선 비율
sum_pct = 0 ## 퍼센티지
bbox_props = dict(boxstyle='square',fc='w',ec='w',alpha=0) ## annotation 박스 스타일

```

① 데이터 준비

```

## annotation 설정
config = dict(arrowprops=dict(arrowstyle='-' ),bbox=bbox_props,va='center')

for i,l in enumerate(labels):
    ang1, ang2 = ax.patches[i].theta1, ax.patches[i].theta2 ## 파이의 시작 각도와 끝 각도
    center, r = ax.patches[i].center, ax.patches[i].r ## 원의 중심 좌표와 반지름길이

    if i < len(labels)-1:
        text = f'{frequency[i]}\n({frequency[i]/total*100:.2f}%)'
    else: ## 마지막 파이 조각은 안그리기
        text = ""

    ## 비율 상한선보다 작은 것들은 Annotation으로 만든다.
    if frequency[i]/total*100 > threshold:
        x = (r/1.5)*np.cos(np.pi/180*((ang1+ang2)/2)) + center[0] ## 텍스트 x좌표
        y = (r/1.5)*np.sin(np.pi/180*((ang1+ang2)/2)) + center[1] ## 텍스트 y좌표
        ax.text(x,y,text,ha='center',va='center',fontsize=13)

## 제목, 범례 추가
plt.title("시도별 게시글", fontsize=16, fontweight='bold',y=0.96)
plt.legend(pie[0],labels,loc=(1.0,0.2), fontsize=11) ## 범례
plt.show()

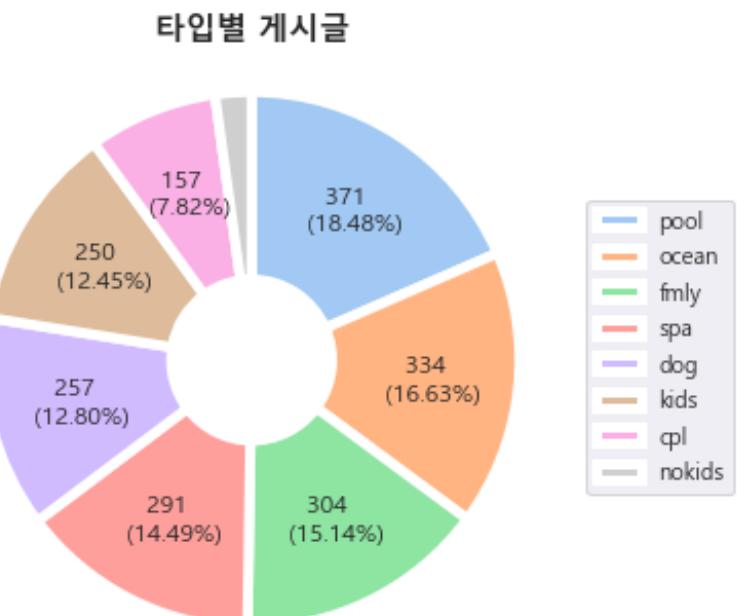
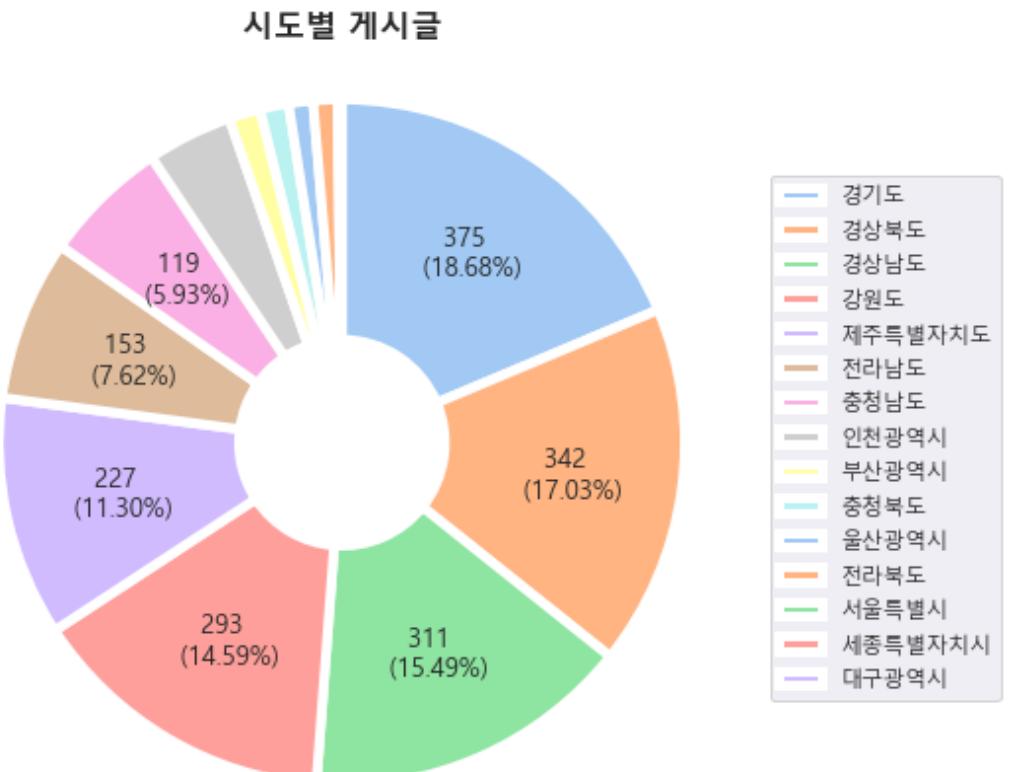
```

② 파이차트 그리기

③ annotation 설정

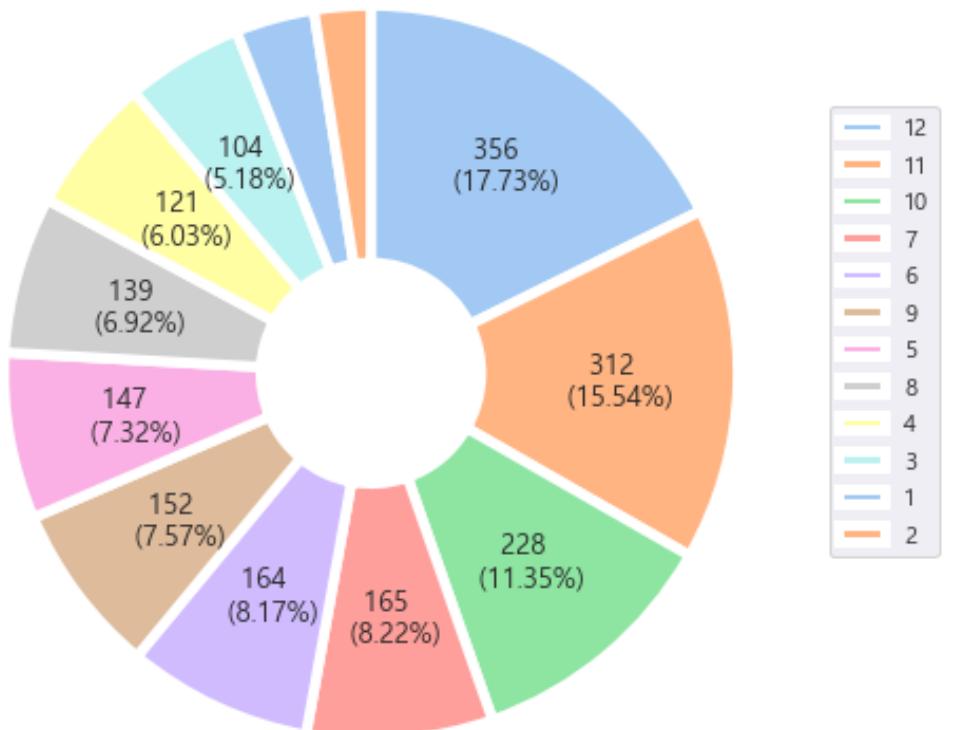
④ 제목, 범례 추가

Visualization_Seaborn 활용

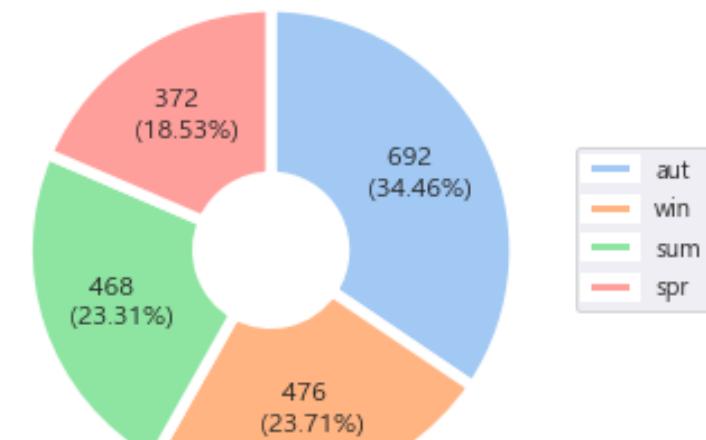


Visualization_Seaborn 활용

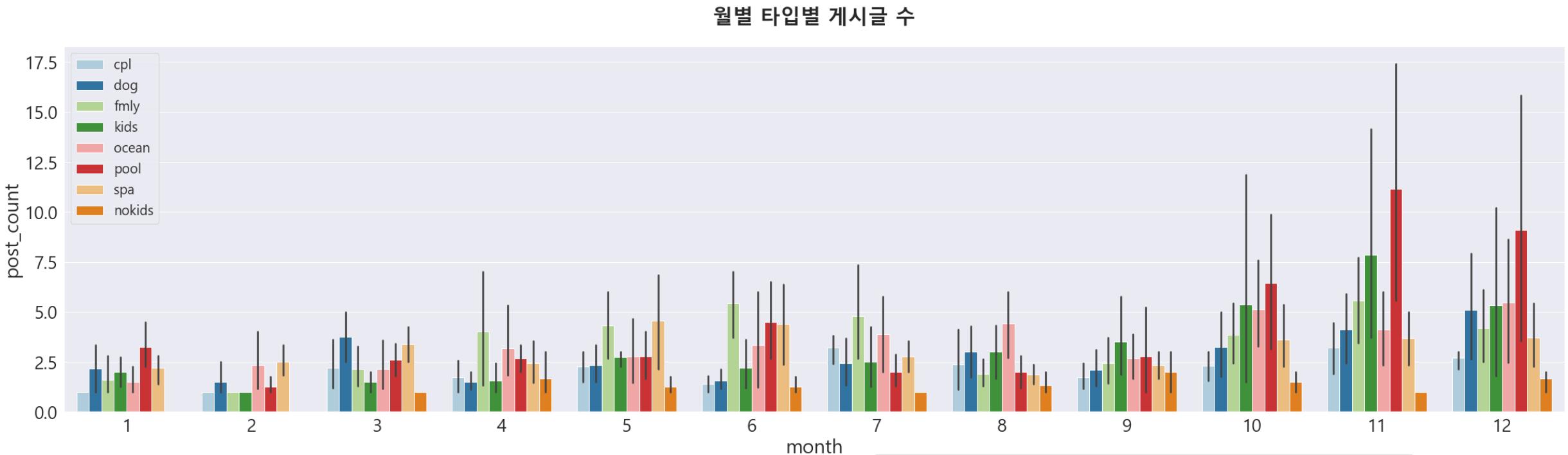
월별 게시글



계절별 게시글



Visualization_Seaborn 활용



```

## 월별 게시글 수(타입으로 색깔 구분) (#★)
fig,ax = plt.subplots(figsize=(40,10))

sns.set(font_scale = 2.4)
sns.set_palette("Paired",n_colors=8, color_codes=True)

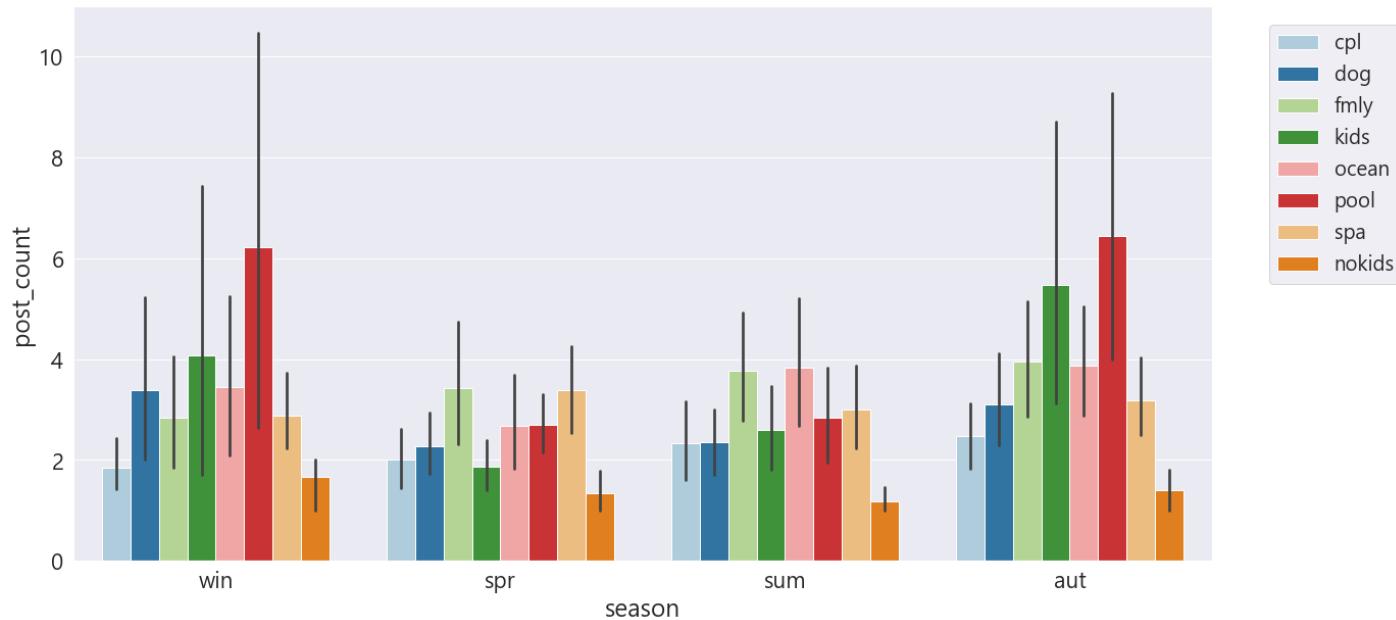
sns.barplot(x='month', y='post_count', hue='type',
            data=p_group, ax=ax);

plt.rcParams['font.family'] = 'Malgun Gothic'
plt.legend(loc='upper left', fontsize=20)
plt.title("월별 타입별 게시글 수", fontweight='bold', fontsize=30,
          loc='center', y=1.05);

```

Visualization_Seaborn 활용

계절별 타입별 게시글 수



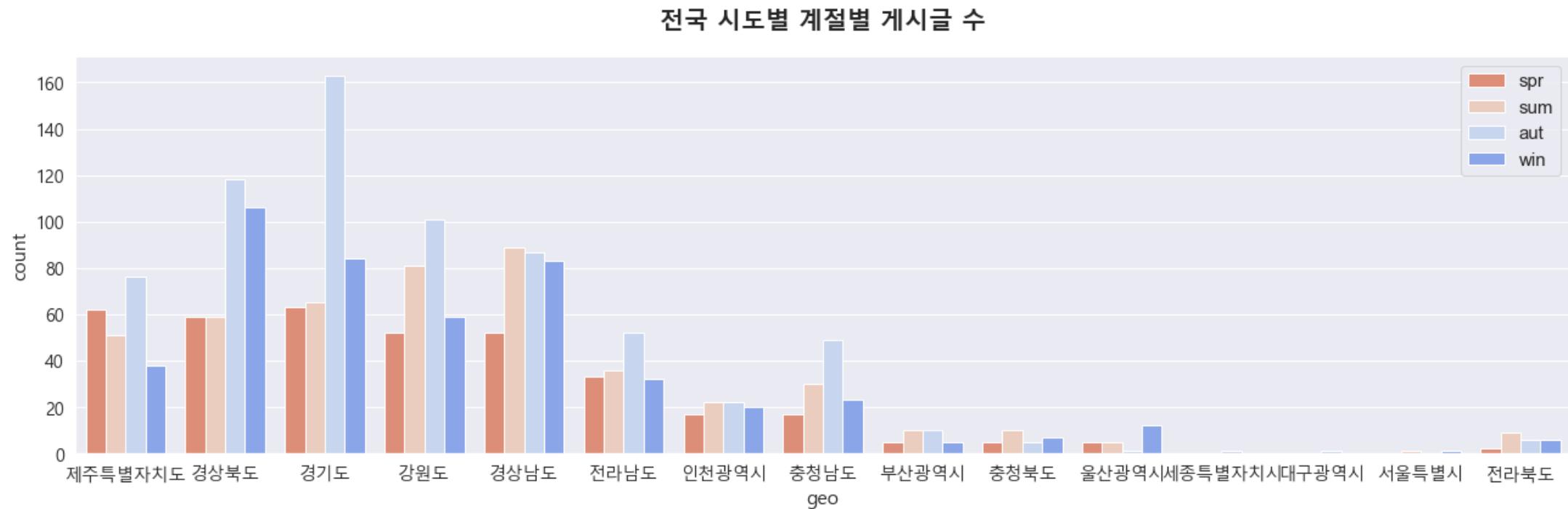
```
## 계절별 게시글 수(타입으로 색깔 구분) ##★
fig,ax = plt.subplots(figsize=(20,10))

sns.set(font_scale = 2.0)
sns.set_palette("Paired",n_colors=8, color_codes=True)

sns.barplot(x='season', y='post_count', hue='type',
            data=p_group, ax=ax);

plt.rcParams['font.family'] = 'Malgun Gothic'
plt.legend(loc=(1.05,0.5), fontsize=20)
plt.title("계절별 타입별 게시글 수", fontweight='bold', fontsize=30,
          loc='center', y=1.05);
```

Visualization_Seaborn 활용



```
## 시도별 게시글 수(계절로 색깔 구분) ##★
fig,ax = plt.subplots(ncols=1, figsize=(22,6))

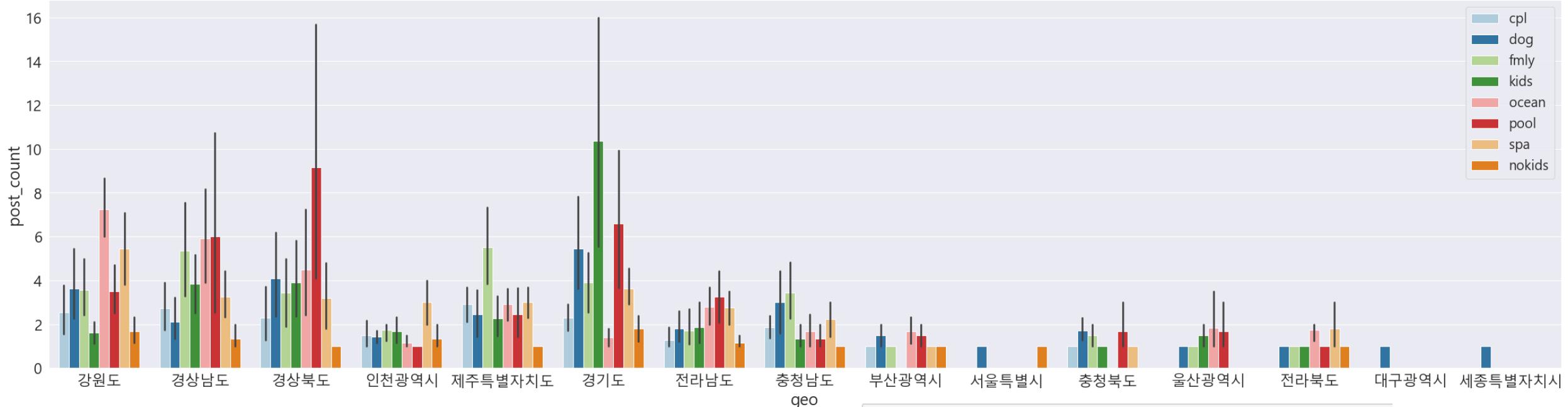
sns.set(font_scale = 1.0)
sns.set_palette("coolwarm_r", n_colors=4, color_codes=True)

sns.countplot(x='geo', hue='season', data=pension_2021,
               hue_order=['spr','sum','aut','win'], ax=ax);

plt.title("전국 시도별 계절별 게시글 수", fontweight='bold',
          fontsize=20, y=1.05)
plt.legend(loc='upper right', fontsize=15)
plt.rcParams['font.family'] = 'Malgun Gothic'
```

Visualization_Seaborn 활용

시도별 타입별 게시글 수



```
## 시도별 게시글 수(타입으로 색깔 구분) ##★
fig,ax = plt.subplots(figsize=(40,10))

sns.set(font_scale = 2.2)
sns.set_palette("Paired",n_colors=8, color_codes=True)

sns.barplot(x='geo', y='post_count', hue='type',|
            data=p_group, ax=ax);

plt.rcParams['font.family'] = 'Malgun Gothic'
plt.legend(loc="upper right", fontsize=20)
plt.title("시도별 타입별 게시글 수", fontweight='bold', fontsize=33,
          loc='center', y=1.05);
```

Visualization_Folium 활용(1)

① 전국 시도 json 데이터 읽어오기

```
## 전국 시도 json 데이터 읽어오기

import json

with open('TL_SCCO_CTPRVN.json', 'r', encoding='utf-8') as file:
    geo_data = json.load(file)
```

② 전국 시도별 게시글 수 히트 맵 생성

```
## 시도별 게시글 히트맵

m_heat = folium.Map([35.91037732354556, 127.88345455043314], zoom_start=7, height=700)

folium.Choropleth(geo_data=geo_data,
                  name='pension_heatmap',
                  data=pension_div,
                  columns=['ctp_name', 'post_count'],
                  key_on='feature.properties.CTP_KOR_NM',
                  fill_color='YIGn',
                  fill_opacity=0.7,
                  line_opacity=0.5,
                  legend_name='Post Counts (%)'
                  ).add_to(m_heat)

m_heat
```

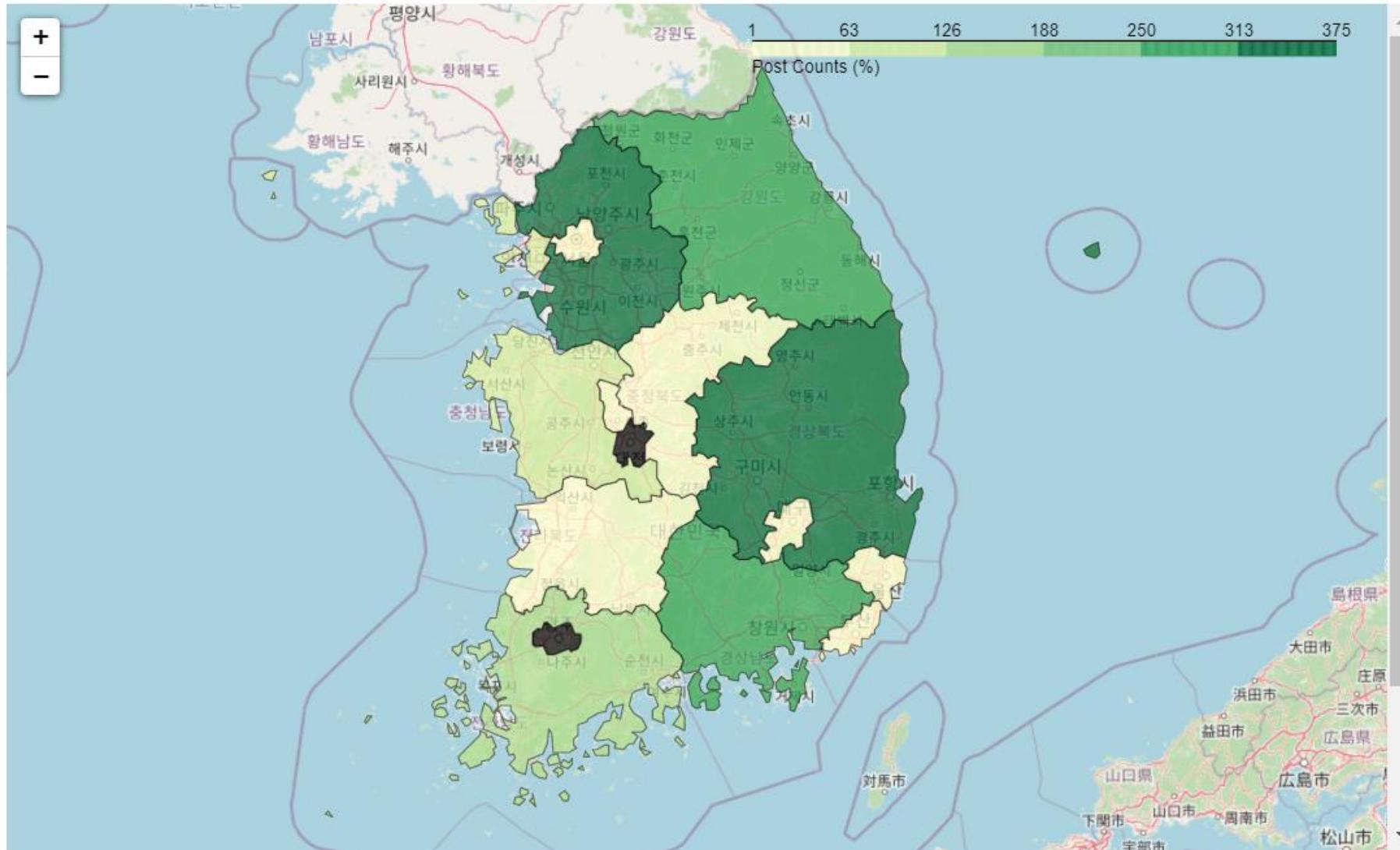
* JSON 이란?

Java Script Object Notation의 약자로
Javascript 객체 문법으로 구조화된 데이터이다.

Python에서는 json 파일을 불러와
Dictionary 객체 형태로 활용할 수 있다.

활용된 json 데이터는 전국의 특별(자치)시, 광역시 및 도를
위도, 경도 데이터로 구분하여 지도모양으로 형성한 파일이다.

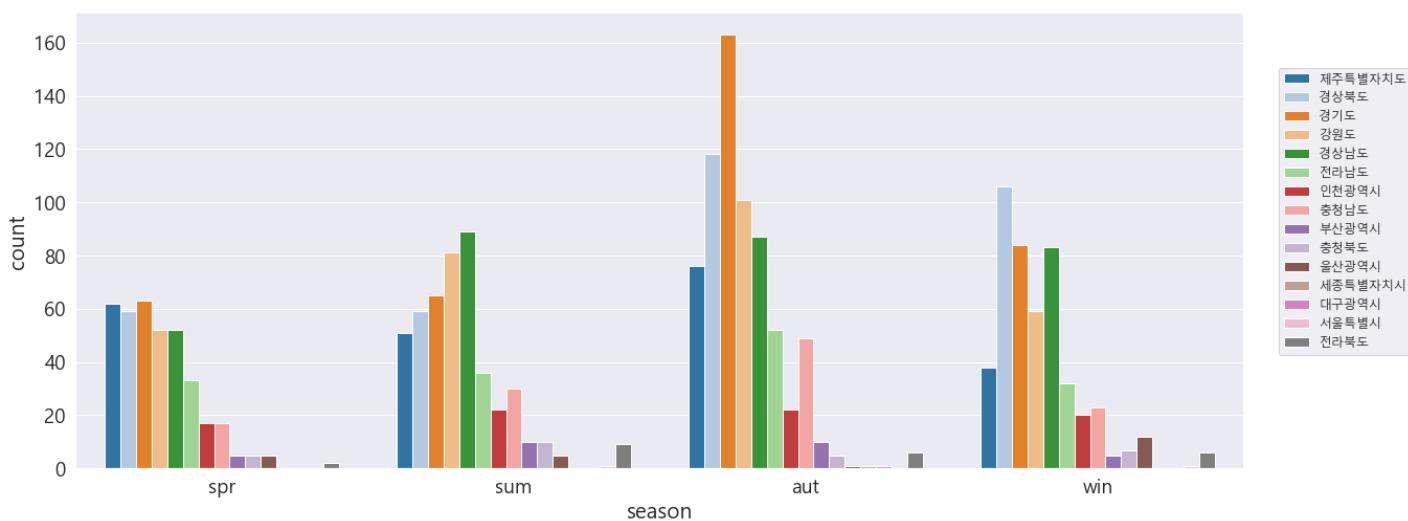
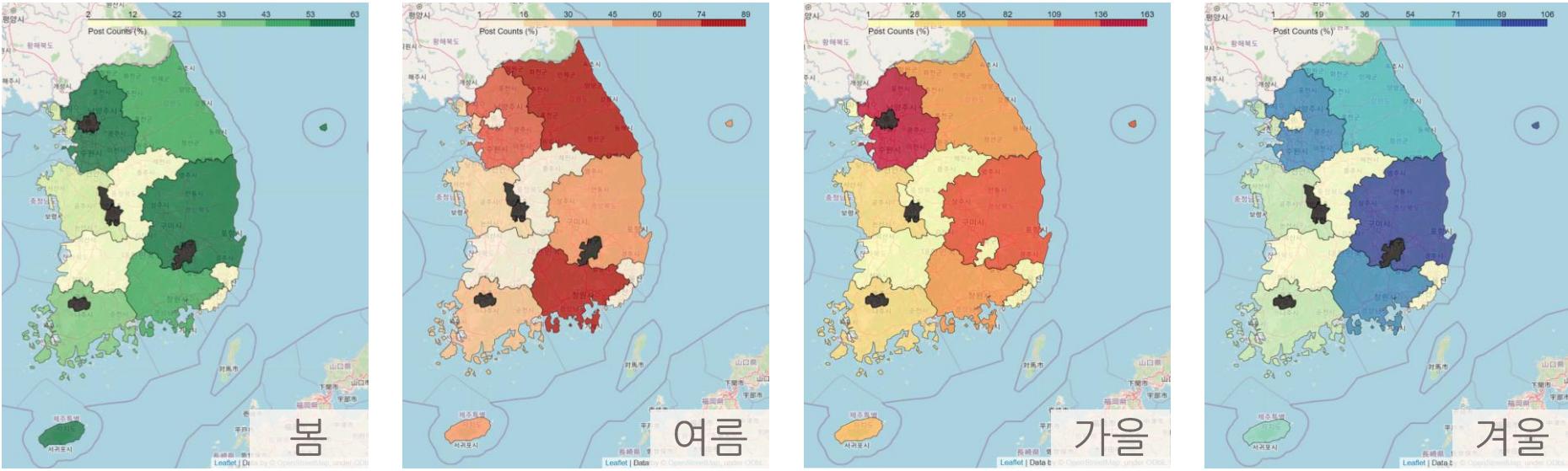
Visualization_Folium 활용(1)



전국 펜션 지도 (1)

- * 지도 구분 = 전국 시도 15개
- * 색깔 = 게시글 수(진할 수록 게시글 수 많음)

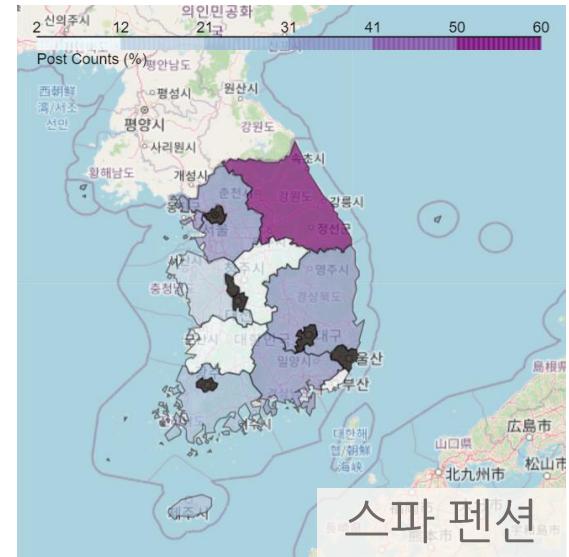
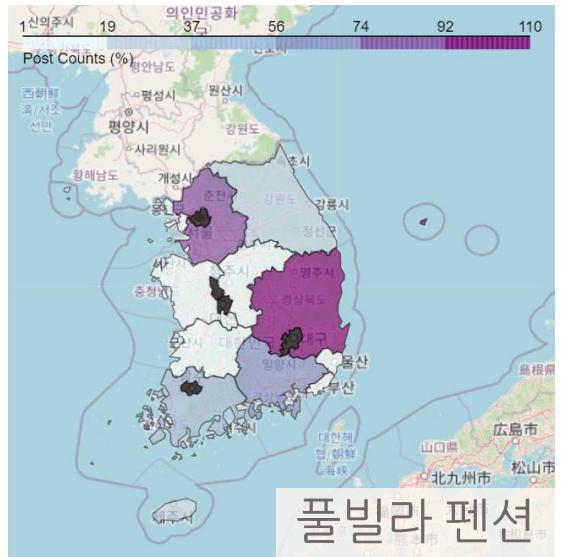
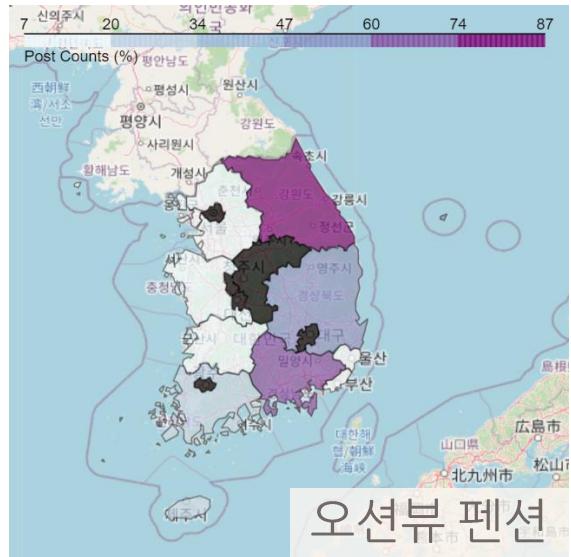
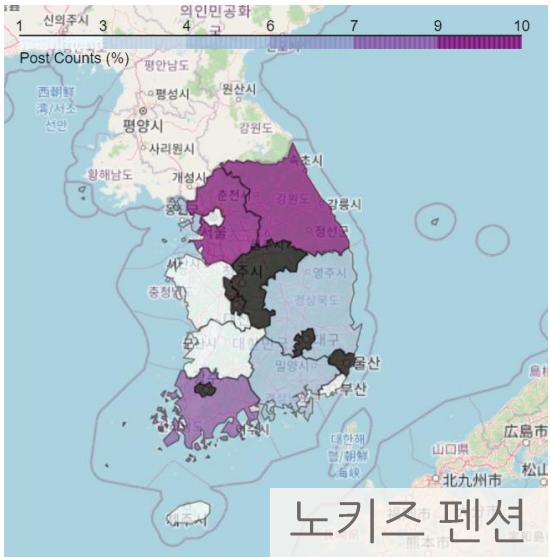
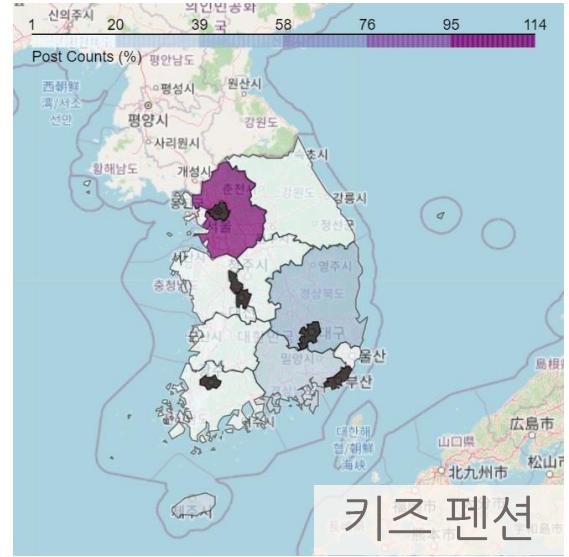
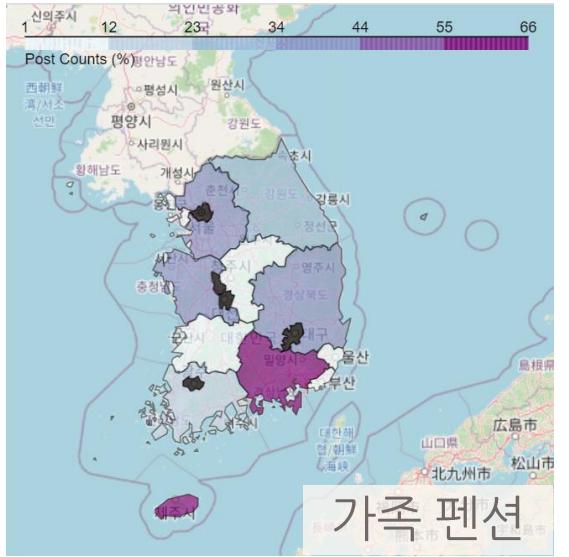
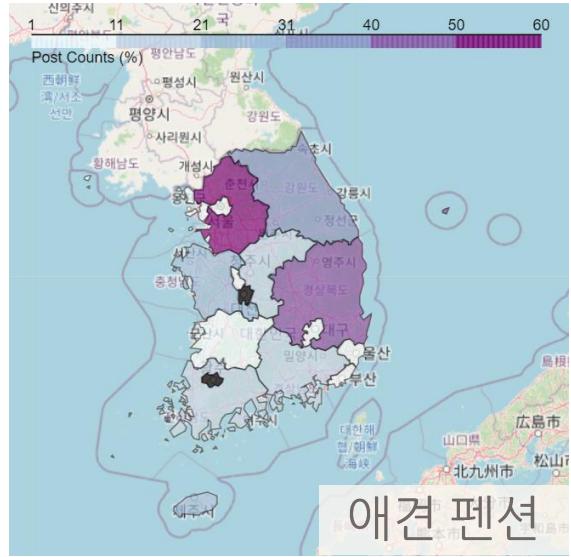
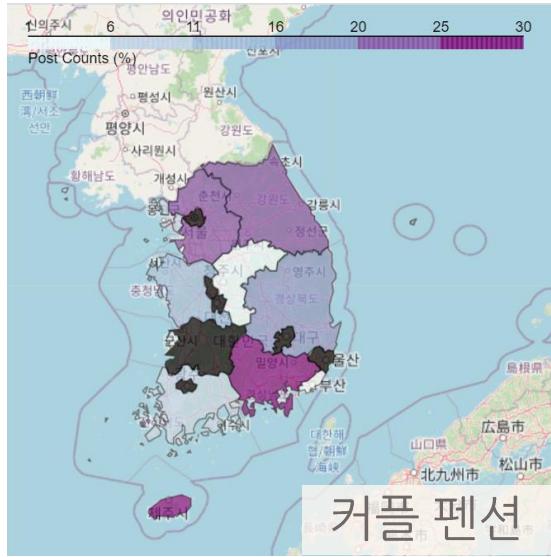
Visualization_Folium 활용(1)



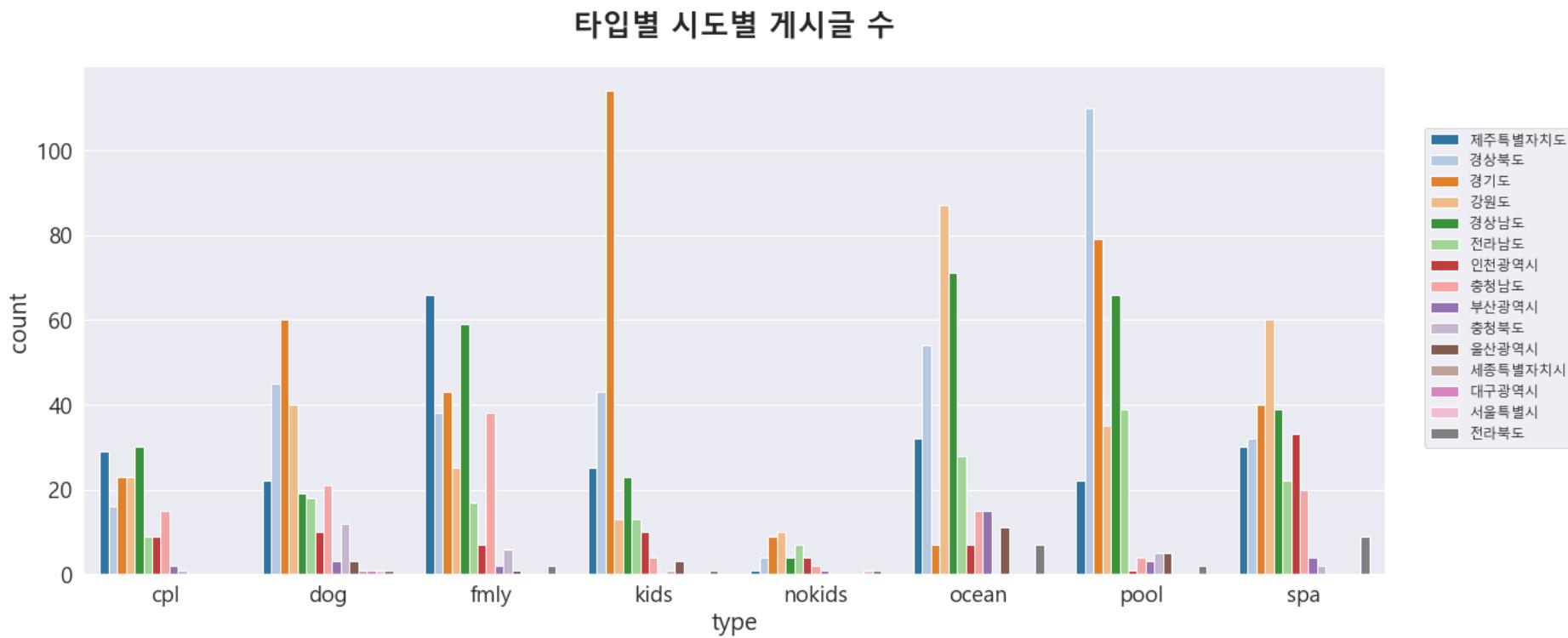
Bar chart와 비교(1)

: 계절별 시도별 게시글 수

Visualization_Folium 활용(1)



Visualization_Folium 활용(1)



Bar chart와 비교(2)

: 펜션 타입별 시도별 게시글 수

Visualization_Folium 활용(2)

```

## 기본지도
korea = [35.61037732354556, 127.88345455043314]
p_map_all = folium.Map(korea, zoom_start=7, height=610)

## 펜션 색깔 지정
color_match = {'cpl':'red','fmlly':'green','kids':'beige','nokids':'purple',
               'spa':'pink','pool':'orange','ocean':'blue','dog':'darkblue'}

for i in pension_2021.index:
    if pension_2021.loc[i,'type'] == 'cpl':
        match = color_match['cpl']
    if pension_2021.loc[i,'type'] == 'fmlly':
        match = color_match['fmlly']
    if pension_2021.loc[i,'type'] == 'kids':
        match = color_match['kids']
    if pension_2021.loc[i,'type'] == 'nokids':
        match = color_match['nokids']
    if pension_2021.loc[i,'type'] == 'spa':
        match = color_match['spa']
    if pension_2021.loc[i,'type'] == 'pool':
        match = color_match['pool']
    if pension_2021.loc[i,'type'] == 'ocean':
        match = color_match['ocean']
    if pension_2021.loc[i,'type'] == 'dog':
        match = color_match['dog']

```

① 기본 맵 생성

② 펜션 타입별 색깔 지정

③ 펜션 위치(위도, 경도) 설정 및 팬수(level)를 원 반지름으로 지정

```

sub_lat = pension_2021.loc[i,'lats']
sub_long = pension_2021.loc[i,'lons']
level = pension_2021.loc[i,'level']*2.5 #팬수 레벨을 반지름 크기로

```

popup 창 만들기

```

name = pension_2021.loc[i,'place_names']
tel = pension_2021.loc[i,'place_tels']
place_url = 'https://map.naver.com/v5/entry/place/' + str(int(pension_2021.loc[i,'place_ids']))
adds = pension_2021.loc[i,'place_addrs']

html = folium.Html(f'''<a href={place_url} target='_blank'>
    {name} </a>
    <p> {tel} </p>
    <p> {adds} </p>
    ...
    script=True)

```

④ 마우스 클릭 시 pop-up 창 생성

```

pop = folium.Popup(html, max_width=250)

```

CircleMarker 만들기

```

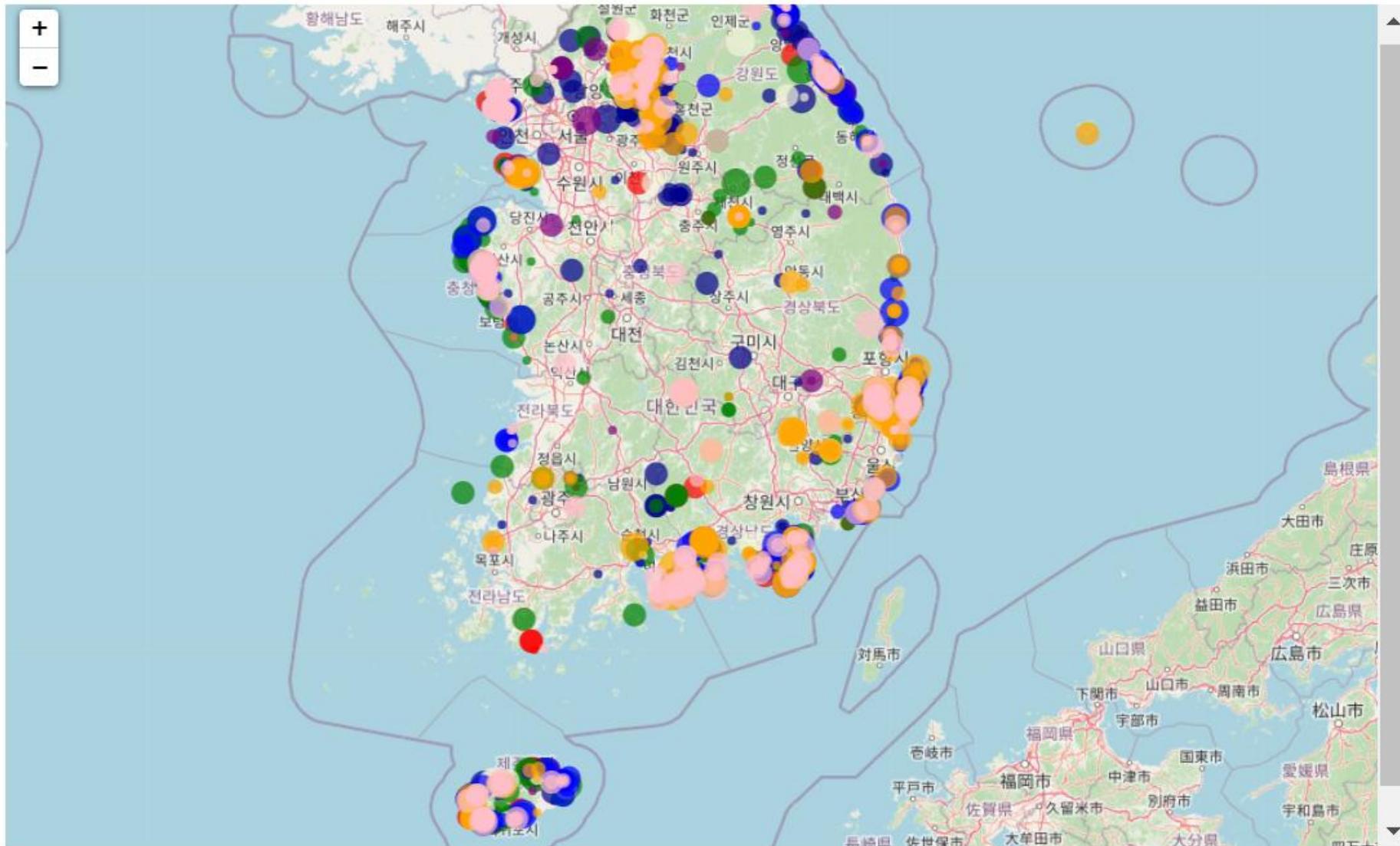
folium.CircleMarker([sub_lat,sub_long], radius=level, weight = 0.0, fill_color=match,
                    fill_opacity = 0.7, popup=pop,).add_to(p_map_all)

```

p_map_all

⑤ 지도 위에 circle marker 생성

Visualization_Folium 활용(2)



전국 펜션 지도 (1)

- * 원 색깔 = 펜션 타입
- * 반지름 크기 = 팬수 level

Visualization_Folium 활용(2)

```

## 기본지도
p_map_cluster = folium.Map([35.81037732354556, 127.88345455043314], zoom_start=7, height=700)
marker_cluster = MarkerCluster().add_to(p_map_cluster)

## 펜션 타입별 색깔 지정
color_match = {'cpl': 'red', 'fmly': 'green', 'kids': 'beige', 'nokids': 'purple',
               'spa': 'pink', 'pool': 'orange', 'ocean': 'blue', 'dog': 'darkblue'}

for i in pension_2021.index:
    if pension_2021.loc[i, 'type'] == 'cpl':
        match = color_match['cpl']
    if pension_2021.loc[i, 'type'] == 'fmly':
        match = color_match['fmly']
    if pension_2021.loc[i, 'type'] == 'kids':
        match = color_match['kids']
    if pension_2021.loc[i, 'type'] == 'nokids':
        match = color_match['nokids']
    if pension_2021.loc[i, 'type'] == 'spa':
        match = color_match['spa']
    if pension_2021.loc[i, 'type'] == 'pool':
        match = color_match['pool']
    if pension_2021.loc[i, 'type'] == 'ocean':
        match = color_match['ocean']
    if pension_2021.loc[i, 'type'] == 'dog':
        match = color_match['dog']

```

② marker cluster 생성

③ 펜션 타입별 색깔 지정

① 기본 맵 생성

- * 원 색깔 = 펜션 타입
- * 반지름 크기 = 팬수 level

④ 펜션 위치(위도, 경도) 설정 및 팬수(level)를 원 반지름으로 지정

```

sub_lat = pension_2021.loc[i, 'lats']
sub_long = pension_2021.loc[i, 'longs']
level = pension_2021.loc[i, 'level'] * 2.5 # 팬수 레벨을 반지름 크기로

## popup 창 만들기
name = pension_2021.loc[i, 'place_names']
tel = pension_2021.loc[i, 'place_tels']
place_url = 'https://map.naver.com/v5/entry/place/' + str(int(pension_2021.loc[i, 'place_ids']))
html = folium.Html(f'''<p> {name} </p>
                    <p> {tel} </p>
                    <a href={place_url}> 네이버 지도 바로가기 </a>
                    ''', script=True)
pop = folium.Popup(html, max_width=250)

```

⑤ 마우스 클릭 시 pop-up 창 생성

```

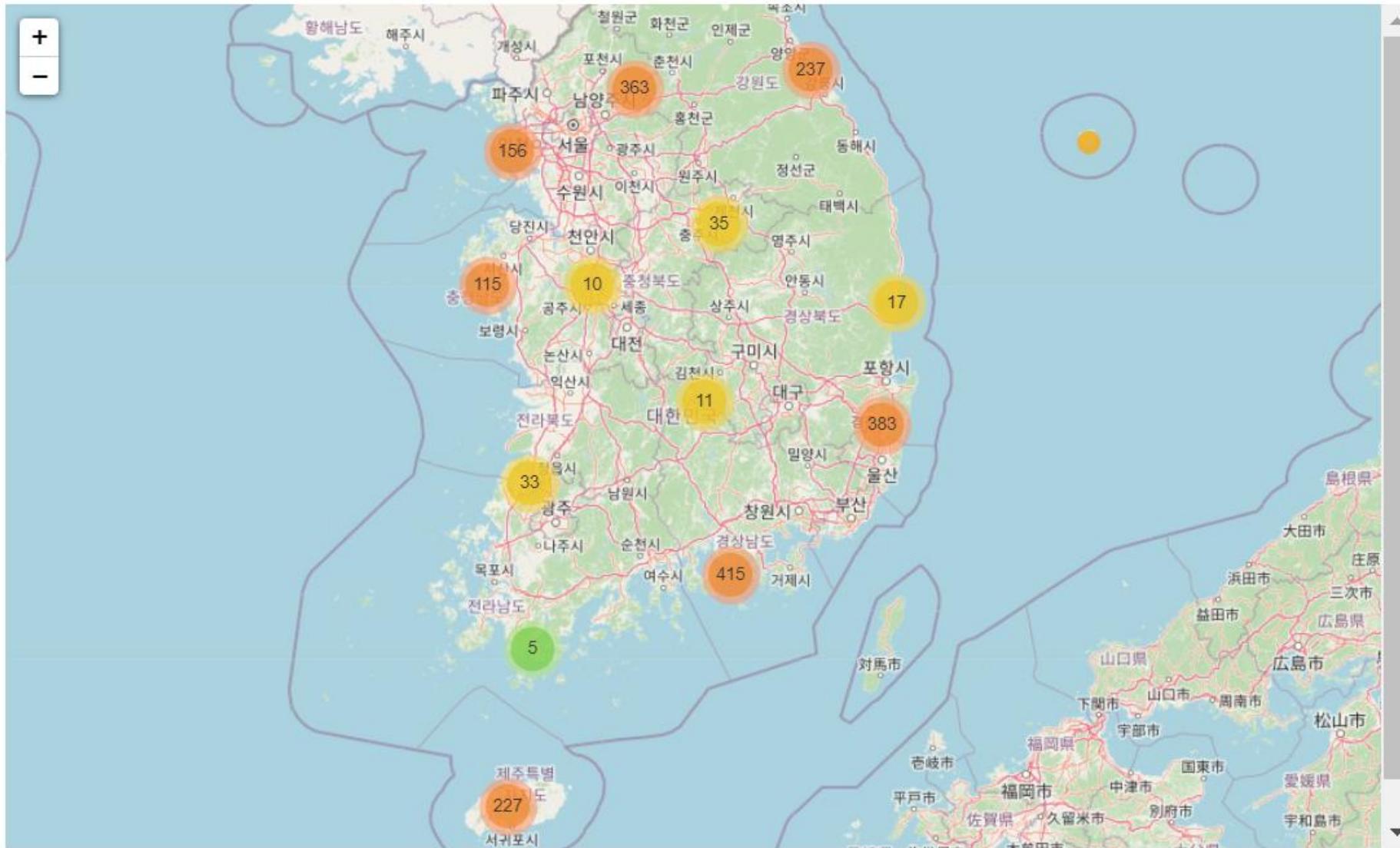
## CircleMarker 만들기
folium.CircleMarker([sub_lat, sub_long], radius=level,
                    weight=0.0, fill_color=match,
                    fill_opacity=0.7, popup=pop).add_to(marker_cluster)

```

p_map_cluster

⑥ 지도 위에 circle marker 생성

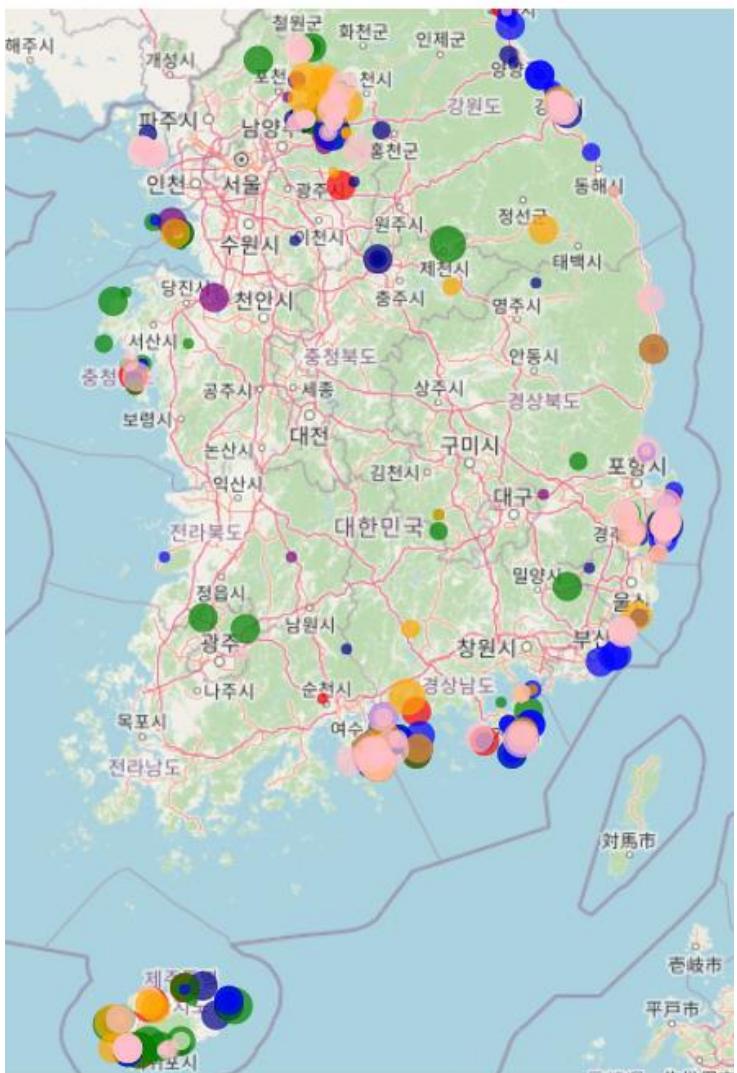
Visualization_Folium 활용(2)



전국 펜션 지도 (2)

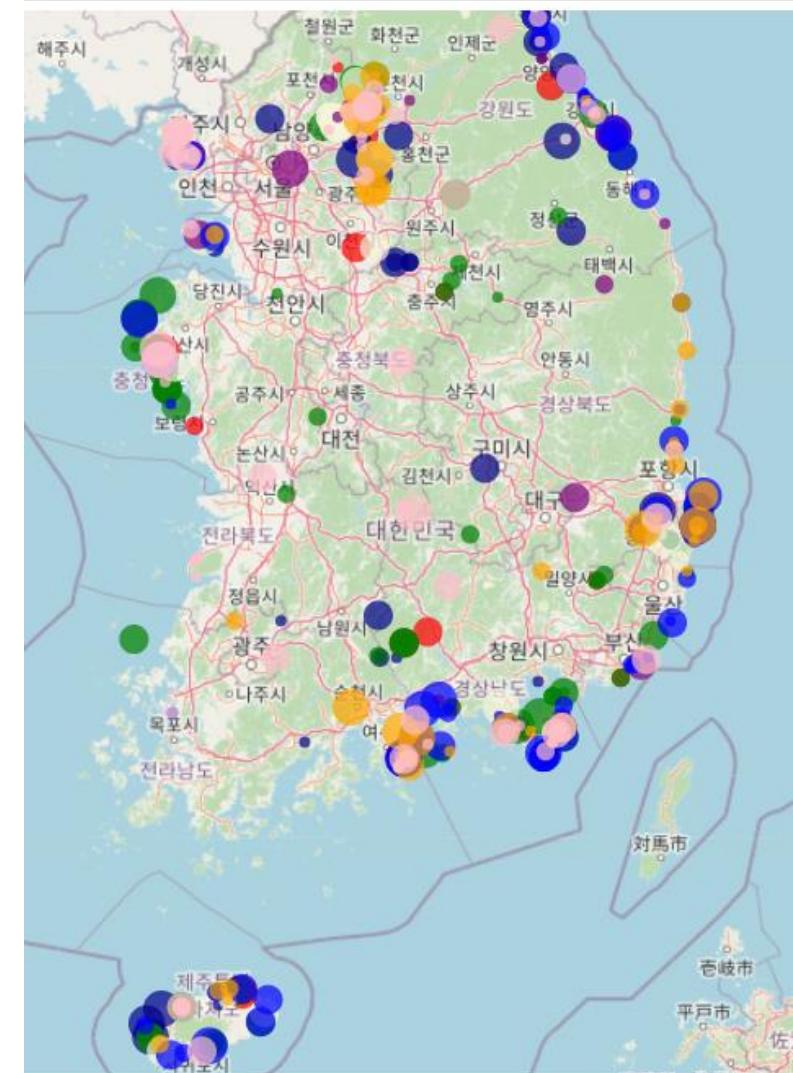
- * 원 색깔 = 펜션 타입
- * 반지름 크기 = 팬 수 레벨
- * 가까운 지역은 cluster로 묶어서 보여주기

Visualization_Folium 활용(2)



전국 펜션 지도 (3)

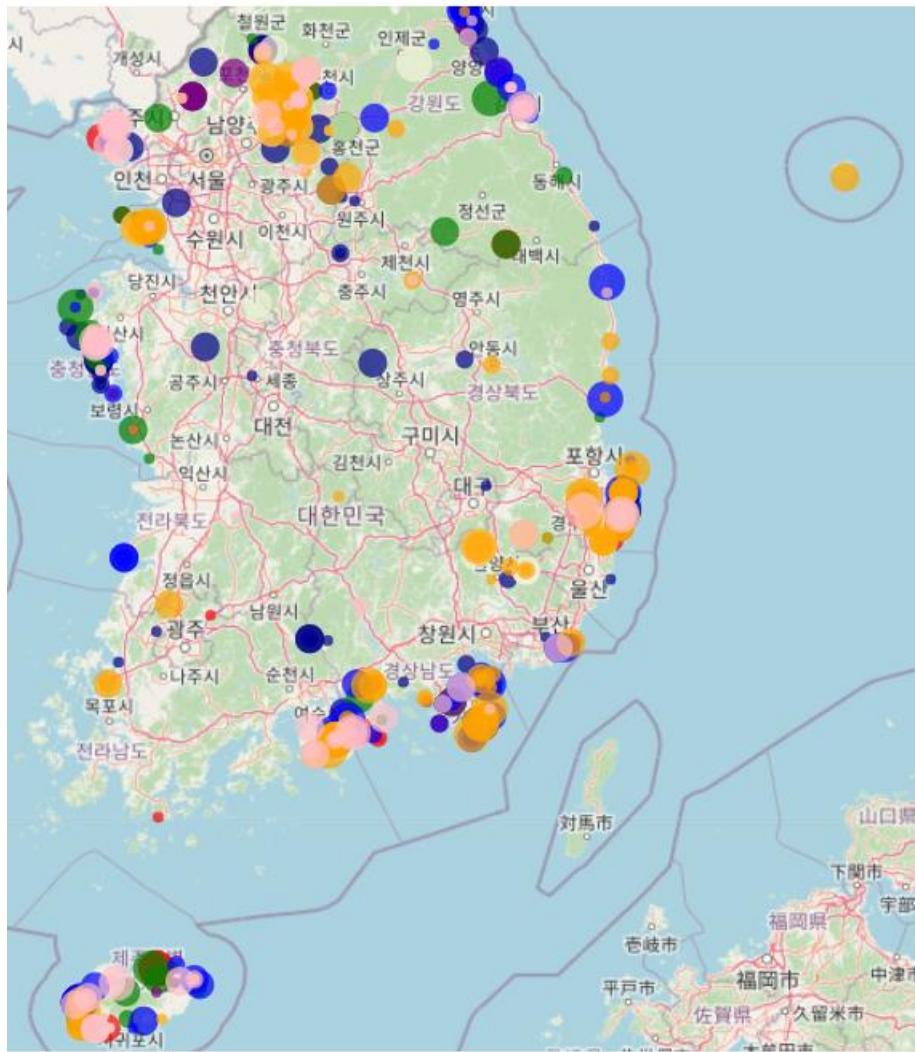
- * 원 색깔 = 펜션 타입
- * 반지름 크기 = 팬 수 level
- * 계절이 봄인 데이터만 활용



전국 펜션 지도 (4)

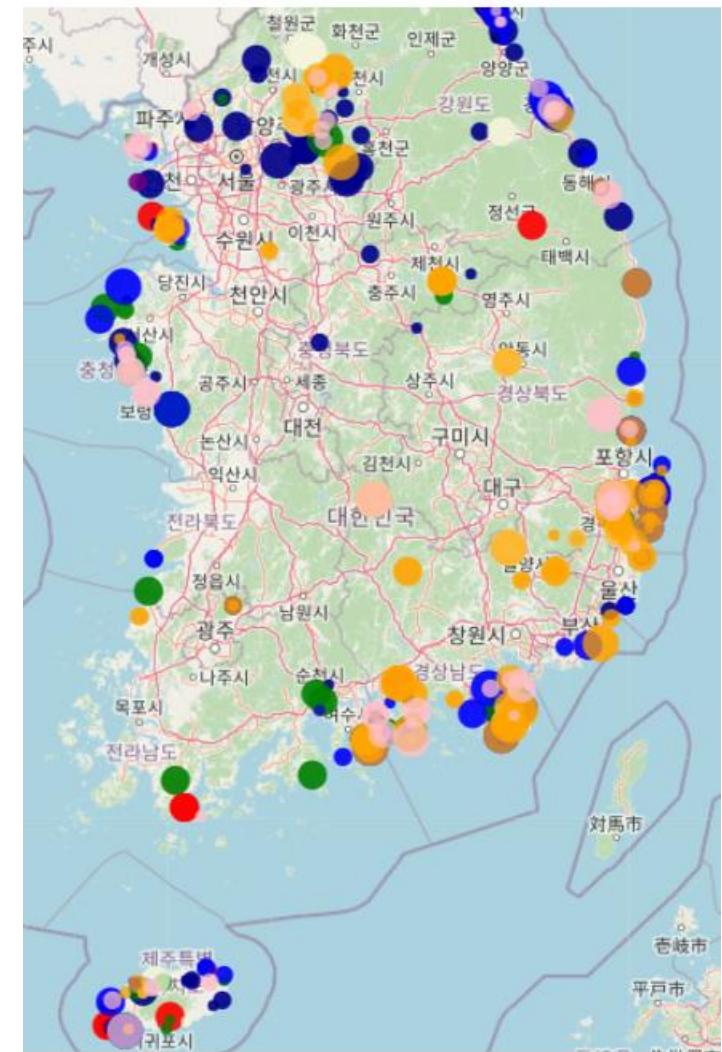
- * 원 색깔 = 펜션 타입
- * 반지름 크기 = 팬 수 level
- * 계절이 여름인 데이터만 활용

Visualization_Folium 활용(2)



전국 펜션 지도 (5)

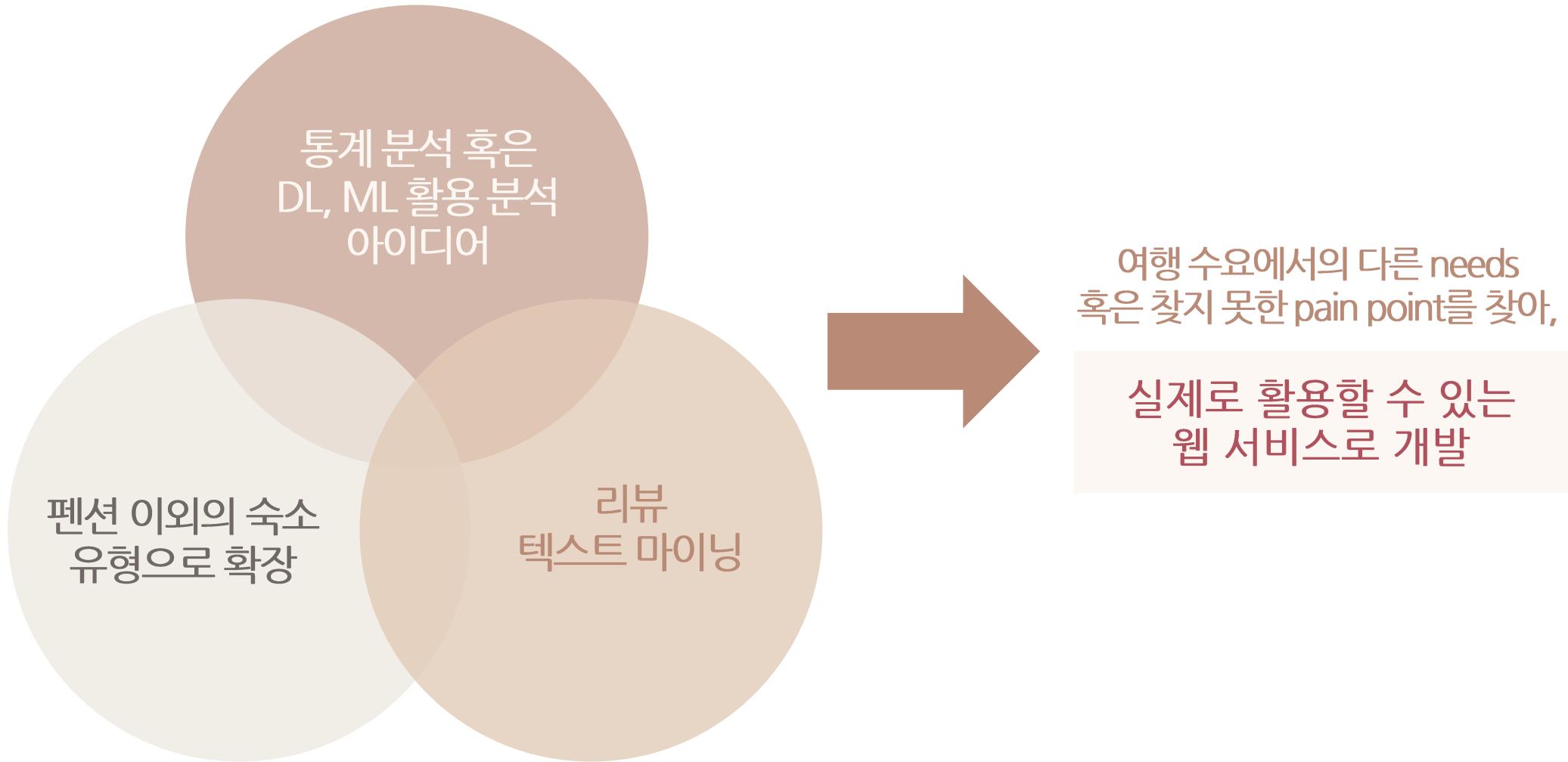
- * 원 색깔 = 펜션 타입
- * 반지름 크기 = 팬수 level
- * 계절이 가을인 데이터만 활용



전국 펜션 지도 (6)

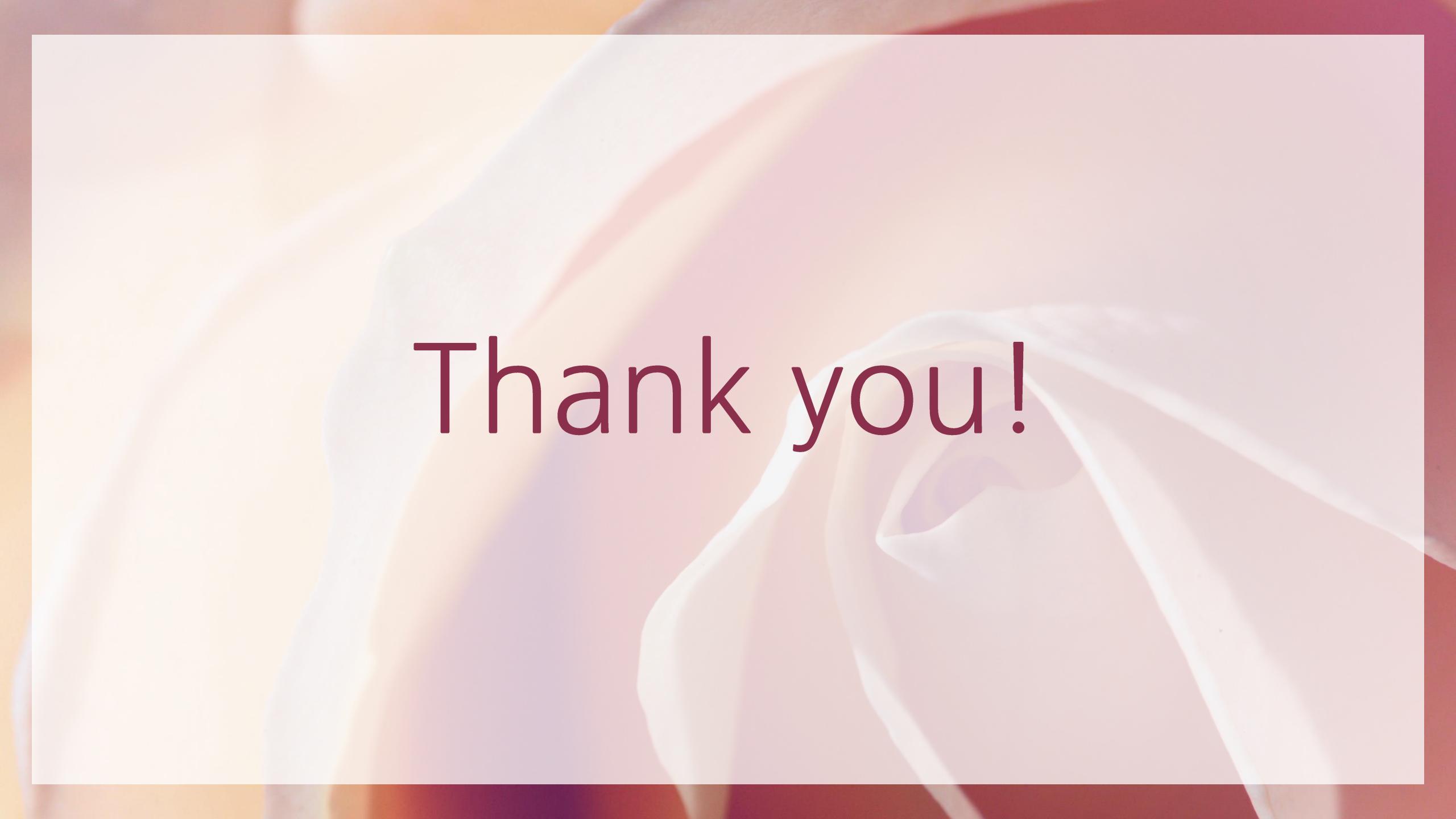
- * 원 색깔 = 펜션 타입
- * 반지름 크기 = 팬수 level
- * 계절이 겨울인 데이터만 활용

More ideas…





Q&A



Thank you!