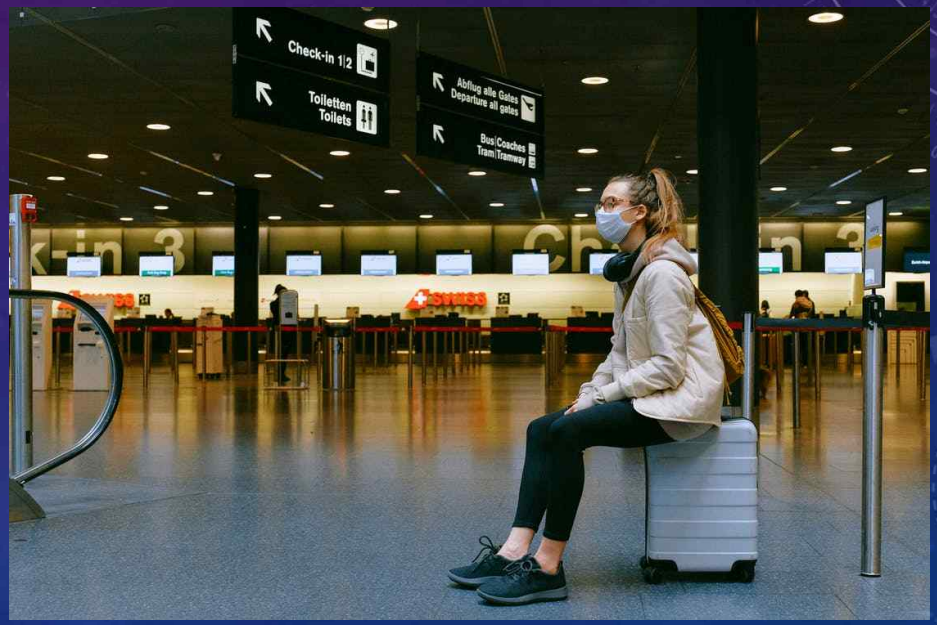
COIVD 19 이후의 여행 관심 지역

이세혁

목차

1. 프로젝트 목표

- 2019년과 2020년을 비교하여 코로나로 인한 여행 관심 지역 변화 파악
- 코로나 이후 관심이 높아진 지역을 파악하여 분석
- 향후 대처 방안



```python
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from bs4 import BeautifulSoup
from konlpy.tag import *
from PIL import Image
from wordcloud import WordCloud
from collections import Counter
from googletrans import Translator

import numpy as np
import matplotlib.pyplot as plt
import chromedriver_autoinstaller
import calendar
import random
import math
import time
import json
```

```python
#크롬 드라이버
chromedriver_autoinstaller.install()
options = webdriver.ChromeOptions()
driver = webdriver.Chrome(options=options)

#변수
kkma = Kkma()

naverDict = dict()
naverWordDict = dict()

xPathList = list()
koreanCityList = list()
countryList = list()
popularCityList = list()
popularCityKoList = list()

basePath = "C:/py_temp/temp"
```

```python
#크롬 드라이버로 url 주소 실행
def InitWeb(url = "", isMaximized = False) :
    if isMaximized :
        driver.maximize_window()

    driver.get(url)
    driver.implicitly_wait(10)

#return : soup
def GetSoup() :
    html = driver.page_source
    return BeautifulSoup(html, "html.parser")

#xPath 주소창에 key 검색 후 enter
def Search(xPath, key) :
    element = driver.find_element_by_xpath(xPath)
    element.click( )
    element.send_keys(key)
    element.send_keys("\n")
    driver.implicitly_wait(10)

#xPath 순차 클릭
def Click(xPathList) :
    for i in xPathList :
        driver.find_element_by_xpath(i).click()
        time.sleep(random.uniform(0.5, 1))

#cnt(원하는 검색 수)와 cntPerPg(한 페이지당 나오는 수)를 이용해 pgCnt(필요한 페이지 수) 계산
#pgCnt 만큼 END 키
def PageDown(cnt, cntPerPg) :
    pgCnt = math.ceil(cnt / cntPerPg)
    for i in range(0, pgCnt + 3) :
        element = driver.find_element_by_tag_name("body")
        element.send_keys(Keys.END)
        time.sleep(random.uniform(1, 1.2))

#json 형태로 저장/불러오기
#저장위치, 이름, 저장할 데이터, 저장/불러오기
def SaveLoad(name, data = None, w = False) :
    fullPath = "%s/%s.txt" %(basePath, name)
    if w :
        with open(fullPath, "w", encoding = "utf-8") as f :
            f.write(json.dumps(data, ensure_ascii=False, indent=4))
            f.close()
    else :
        with open(fullPath, "r", encoding = "utf-8") as f :
            data = json.load(f)
            f.close()
            return data
```

1. PYTHON CRAWLING

- 사용된 라이브러리 및 메인 함수
- chromedriver_autoinstaller로 크롬 드라이버 실행
- 변수 선언/초기화

2-1. 네이버 블로그

```python
#네이버 날짜 검색
def SetNaverDate(yyyymmdd) :
    date = yyyymmdd.split("-")
    basePath = '//*[@id="snb"]/div[2]/ul/li[3]/div/div[2]/div[2]/div[YMD]/div/div/div/ul/li[NUM]/a'

    yearPath = basePath.replace("YMD", "1").replace("NUM", str(int(date[0])-2002))
    monthPath = basePath.replace("YMD", "2").replace("NUM", date[1])
    datePath = basePath.replace("YMD", "3").replace("NUM", date[2])

    xPathList.append(yearPath)
    xPathList.append(monthPath)
    xPathList.append(datePath)

#네이버 날짜 월별 1~말일 검색후 마지막 페이지까지
def SearchNaverByMonth(year, month) :
    startDate = "%s-%s-%s" %(year, month, "1")
    endDate = "%s-%s-%s" %(year, month, calendar.monthrange(year, month)[1])

    xPathList.clear()
    xPathList.append('//*[@id="snb"]/div[2]/ul/li[3]/div/div[1]/a[9]')
    SetNaverDate(startDate)
    xPathList.append('//*[@id="snb"]/div[2]/ul/li[3]/div/div[2]/div[1]/span[3]/a')
    SetNaverDate(endDate)
    xPathList.append('//*[@id="snb"]/div[2]/ul/li[3]/div/div[2]/div[3]/button')
    Click(xPathList)

    PageDown(1000, 30)
    contents = GetSoup().find("ul", "lst_total").find_all("a", "api_txt_lines total_tit")

    if len(contents) == 1000 :
        return contents
    else :
        PageDown(1000 - len(contents), 30)
        return contents

#한글명사 추출
def GetNouns(container, data) :
    for i in data :
        nouns = kkma.nouns(i.get_text().strip())
        for j in nouns :
            container.append(j)
```

```python
#네이버 블로그
InitWeb("https://www.naver.com/", True)
Search('//*[@id="query"]', "여행")

xPathList.clear()
xPathList.append('//*[@id="lnb"]/div[1]/div/ul/li[5]/a')
xPathList.append('//*[@id="snb"]/div[1]/div/div[1]/a[2]')
xPathList.append('//*[@id="snb"]/div[1]/div/div[2]/a')

Click(xPathList)

for i in (2019, 2020) :
    naverDict[i] = []
    for j in range(1, 13) :
        titles = SearchNaverByMonth(i, j)
        GetNouns(naverDict[i], titles)

SaveLoad("naver", naverDict, True)
naverDict = SaveLoad("naver")

#====================================
print("2019 :")
print(Counter(naverDict["2019"]).most_common(10))
print("\n2020 :")
print(Counter(naverDict["2020"]).most_common(10))
#====================================
```

```
2019 :
[('여행', 11652), ('일', 1377), ('박', 1222), ('자유', 1207), ('코스', 1197), ('자유여행', 1018), ('제주', 909), ('추천', 900), ('2', 883), ('3', 837)]

2020 :
[('여행', 11702), ('제주', 1669), ('도', 1522), ('코스', 1198), ('곳', 1111), ('일', 931), ('2', 869), ('박', 820), ('1', 806), ('제주도', 761)]
```

- 네이버 블로그 2019, 2020년도 여행 검색 정보 수집
- 여행으로 검색, 월별 데이터 1,000 건 씩 크롤링
- KonIpy를 이용하여 2019년과 2020년 각각 가장 많이 검색된 단어 순서대로 수집

2-2. 나라 및 국내 도시 목록

```python
#나라 목록
InitWeb("https://ko.wikipedia.org/wiki/나라_목록")
time.sleep(0.5)
data = GetSoup().find("div", "mw-body-content mw-content-ltr").find_all("b")

for i in data :
    txt = i.get_text()
    if txt.find(" ") == 0 :
        txt = txt.lstrip(" ")
        countryList.append(txt)

countryList = list(set(countryList))

SaveLoad("country", countryList, True)
countryList = SaveLoad("country")

#===================================
countryList
#===================================
```

```
['엘살바도르',
 '베네수엘라',
 '앤티가 바부다',
 '볼리비아',
 '코모로',
 '요르단',
```

```python
#대한민국 도시 목록
InitWeb("https://ko.wikipedia.org/wiki/대한민국의_인구순_도시_목록")
time.sleep(0.5)
data = GetSoup().find("table", "wikitable sortable jquery-tablesorter").find_all("a")

for i in data :
    txt = i.get_text().rstrip("시").rstrip("군")
    txt = txt.replace("남도", "").replace("북도", "")

    if txt.find("특별") != -1 :
        txt = txt.split("특별")
        txt = txt[0]
    elif txt.find("광역") != -1 :
        txt = txt.split("광역")
        txt = txt[0]

    if txt == "경기도" : txt = "경기"
    if txt == "강원도" : txt = "강원"

    koreanCityList.append(txt)

koreanCityList = list(set(koreanCityList))

SaveLoad("koreanCity", koreanCityList, True)
koreanCityList = SaveLoad("koreanCity")

#===================================
koreanCityList
#===================================
```

```
['보은',
 '장흥',
 '영양',
 '정기',
 '완주',
```

- Wikipedia에서 나라 목록과 국내 모든 도시 목록 크롤링
- 도시 목록에서 시/군/도와 같은 행정구역 단위 제거

2-3. 세계 도시 목록

```python
#유명 도시 목록(영)
InitWeb("https://www.visualcapitalist.com/the-100-most-popular-city-destinations/")
time.sleep(0.5)
driver.find_element_by_xpath("//select[@name='tablepress-1056_length']/option[text()='100']").click()
data = GetSoup().find("table", "tablepress tablepress-id-1056 dataTable no-footer").find_all("tr")

for i in data :
    contents = i.find_all("td")
    if (len(contents) != 0) :
        city = contents[1].get_text()
        popularCityList.append(city)

popularCityList = list(set(popularCityList))

SaveLoad("popularCity", popularCityList, True)
popularCityList = SaveLoad("popularCity")

#==================================
popularCityList
#==================================
```

```
['Kolkatta',
 'Florence',
 'London',
 'Osaka',
 'Jerusalem',
```

```python
#유명 도시 목록(한)
tempList = list()

for i in popularCityList :
    translator = Translator()
    result = translator.translate(i, src="en", dest="ko")
    tempList.append(result.text)

for i in tempList :
    city = i.replace(" ", "").rstrip("도시")

    if city == "JohorBahru." : city = "조호르바루"
    elif city == "Zhuhai." : city = "주하이"
    elif city == "heraklion." : city = "헤라클리온"
    elif city == "무덤" : city = "물라"
    elif city == "심천" : city = "선전"
    elif city == "올랜" : city = "올랜도"
    elif city == "계곡" : city = "구이린"
    elif city == "서울" : continue
    elif city == "제주" : continue

    popularCityKoList.append(city)

SaveLoad("popularCityKo2", popularCityKoList, True)
popularCityKoList = SaveLoad("popularCityKo2")

#==================================
popularCityKoList
#==================================
```

```
['콜카타',
 '피렌체',
 '런던',
 '오사카',
 '예루살렘',
```

- Visual Capitalist에서 여행지로 유명한 세계 도시 Top 100 크롤링
- googletrans 라이브러리를 이용하여 번역 후, 잘못된 번역 수정 및 국내 도시 제거

2-4. 나라 및 도시 색출

```python
#나라 및 도시 색출
def FindPlaces(data) :
    placeList = list()

    for i in data :
        if any(chr.isdigit() for chr in i) : continue
        elif len(i) == 1 : continue

        for j in countryList :
            if i.find(j) != -1 :
                placeList.append(j)
                break

        for j in popularCityList :
            if i.find(j) != -1 :
                placeList.append(j)
                break

        for j in koreanCityList :
            if i.find(j) != -1 :
                placeList.append(j)
                break

    return placeList

for i in ("2019", "2020") :
    naverWordDict[i] = Counter(FindPlaces(naverDict[i]))
    name = "naver%s" %i
    SaveLoad(name, naverWordDict[i], True)

#===================================
print("2019 :")
print(naverWordDict["2019"].most_common(10))
print("\n2020 :")
print(naverWordDict["2020"].most_common(10))
#===================================
```

```
2019 :
[('제주', 1693), ('일본', 455), ('부산', 415), ('강원', 399), ('베트남', 326), ('스페인', 254), ('중국', 247), ('이탈리아', 229), ('강릉', 218), ('미국', 212)]

2020 :
[('제주', 3045), ('강원', 747), ('부산', 470), ('강릉', 431), ('서울', 333), ('경주', 327), ('여수', 299), ('속초', 285), ('거제', 262), ('남해', 194)]
```

```python
#naver 2019, 2020 워드 클라우드
for i in naverWordDict.keys() :
    imgPath = "C:/py_temp/temp/%s.png" %i
    img = np.array(Image.open(imgPath))

    data = dict(naverWordDict[i].most_common(200))
    wordCloud = WordCloud(font_path="C:₩py_temp₩malgun.ttf",
                          relative_scaling=0.2,
                          mask=img,
                          background_color="white",
                          min_font_size=1,
                          max_font_size=50,
                          max_words=500
                         ).generate_from_frequencies(data)

    plt.figure(figsize=(30,20))
    plt.imshow(wordCloud)
    plt.axis("off")
    plt.show()
```

- 수집된 데이터(네이버 블로그)에서 불용어 제거
- 나라 및 도시 단어별 빈도수 집계
- 2019년, 2020년 각각의 해에 맞는 이미지(돼지, 쥐)로 워드 클라우드 생성

3. WORD CLOUD

2019년 (己亥年)

2020년 (庚子年)

```python
import pandas as pd
import json

#path 설정
def GetPath(fileName) :
    basePath = "C:/py_temp/temp"
    return "%s/%s" %(basePath, fileName)

#json 형태로 저장/불러오기
#저장위치, 이름, 저장할 데이터, 저장/불러오기
def LoadJson(path) :
    with open(path, "r", encoding = "utf-8") as f :
        data = json.load(f)
        f.close()
        return data

#param1~2 리스트에 맞춰서 xls 형식으로 저장
#param1 첫행 리스트 / param2 나머지 데이터 리스트 리스트
def SaveCsv(param1, param2, path) :
    df = pd.DataFrame()
    for i in range(0, len(param1)) :
        df[param1[i]]=pd.Series(param2[i])
        df.to_csv(path, index = False, encoding = "utf-8-sig")
```

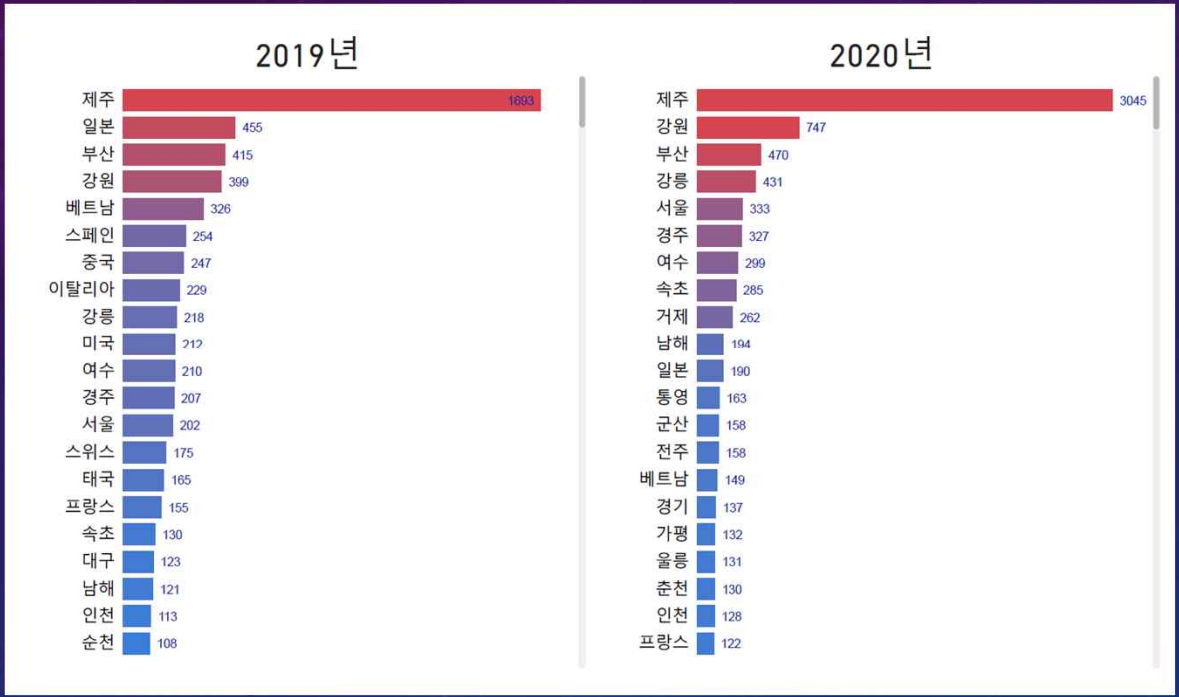
```python
naver = dict()
naver[2019] = LoadJson(GetPath("naver2019.txt"))
naver[2020] = LoadJson(GetPath("naver2020.txt"))

for i in naver.keys() :
    param1 = ["단어", "빈도"]
    param2 = [naver[i].keys(), naver[i].values()]
    SaveCsv(param1, param2, GetPath("naver%s.csv" %i))
```

단어	2019.빈도	2020.빈도	2019.백분율	2020.백분율	백분율 차이
강릉	218	431	52.53	91.7	39.17
거제	92	262	22.17	55.74	33.58
속초	130	285	31.33	60.64	29.31
서울	202	333	48.67	70.85	22.18
경주	207	327	49.88	69.57	19.69
군산	68	158	16.39	33.62	17.23
포항	32	115	7.71	24.47	16.76
울릉	47	131	11.33	27.87	16.55
전주	72	158	17.35	33.62	16.27
통영	77	163	18.55	34.68	16.13
태안	37	116	8.92	24.68	15.77
하동	36	113	8.67	24.04	15.37
경기	58	137	13.98	29.15	15.17
여수	210	299	50.6	63.62	13.01
가평	65	132	15.66	28.09	12.42
고성	56	121	13.49	25.74	12.25
남해	121	194	29.16	41.28	12.12
신안	19	75	4.58	15.96	11.38
서귀포	44	102	10.6	21.7	11.1
동해	30	86	7.23	18.3	11.07
춘천	75	130	18.07	27.66	9.59

4. POWER BI

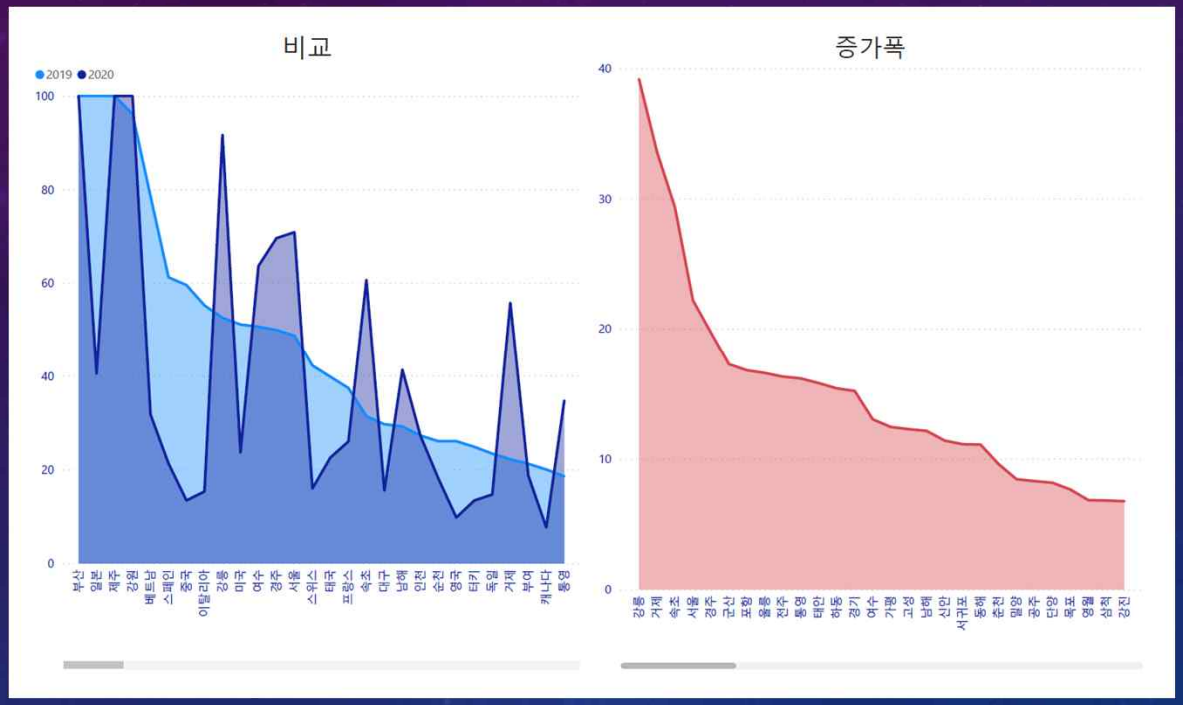
- Json으로 저장된 데이터 csv로 저장
- Power BI 테이블로 가져온 후, 2019년 2020년 데이터 통합

4-1. 여행관심 지역 빈도 비교



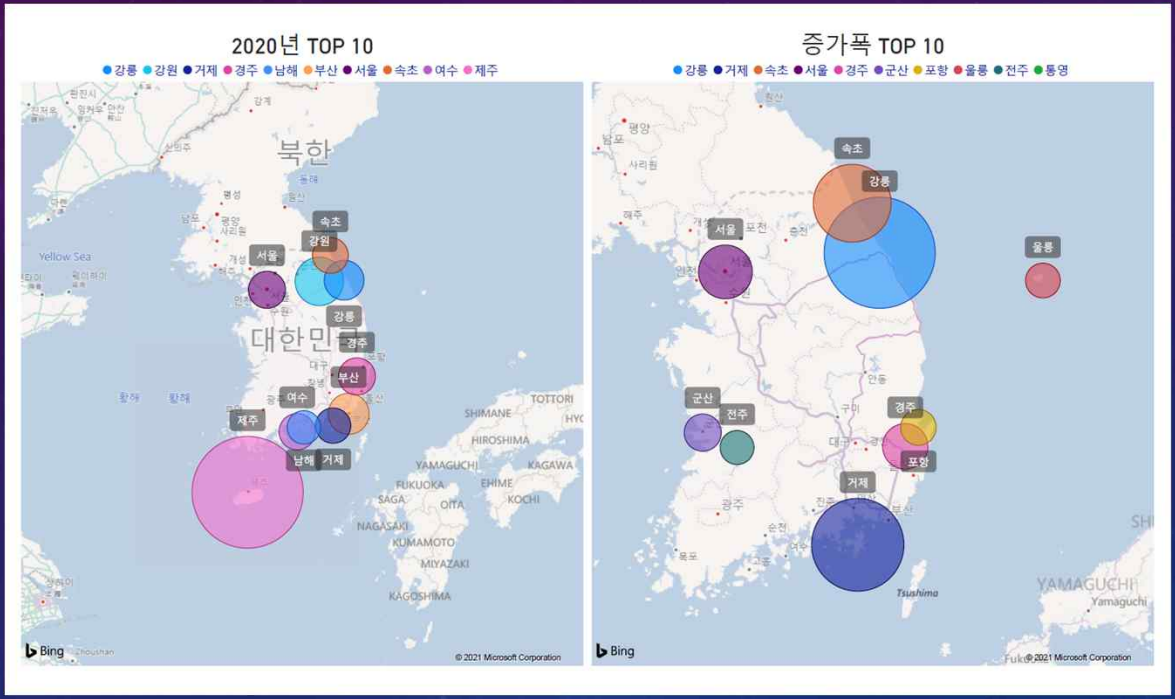
- 코로나로 인하여, 국내여행에 대한 관심이 늘고 해외여행에 대한 관심도가 낮아졌습니다.
- 제주와 강원이 가장 큰 폭으로 증가하였습니다.

4-2. 백분율 비교와 증가 폭



- 2019년과 2020년 모두 빈도수 3위를 차지하였던 부산을 기준(100)으로 하여 데이터를 산출하였습니다.
- 100 이상의 데이터는 100 으로 처리하여, 2020년 새롭게 떠오르는 관심지역을 찾을 수 있었습니다.

4-2. 지도 그래프



- 2020년 여행 관심지역은 모두 국내지역이 차지하였습니다.
- 제주가 가장 관심이 많은 지역이나, 2020년 새롭게 떠오르는 지역은 강릉, 거제, 속초 순이었습니다.

5. 결론

- 2019년 대비 2020년은 여행 관심 지역이 해외에서 국내로 변하였습니다.
- 제주, 부산, 강원은 2019년과 2020년 모두 상위 여행 관심 지역이었습니다.
- 2020년에 새롭게 떠오르는 여행 관심 지역은 강릉, 거제, 속초가 차지하였습니다.
- 코로나가 안정화된 후에도 국내 여행 산업을 유지 활성화하기 위해서는 새롭게 떠오르고 있는 지역의 관광지 개발 및 홍보가 강화되어야 할 것입니다.