GIS with Python！

PyConJP2021 2021/10/15
Chomoku LLC: Hideyuki Ogawa
Translate with DEEPL

自己紹介

Name: Hideyuki Ogawa (@ogawahideyuki)

Company: Chomoku LLC (@hijichomoku)

Founder & CEO

We are hiring!!! Sorry we have no english site. (japanese site)

where: Kyoto

hobby: Jogging / Learn Chinese / BTS / No drinking

Talk: PyConJP2019, PyCon China Beijing 2019, PyCon mini HIroshima, PyConJP 2020 Tutorial

On PyConJP Blog

Python Boot Camp in 京都を開催しました

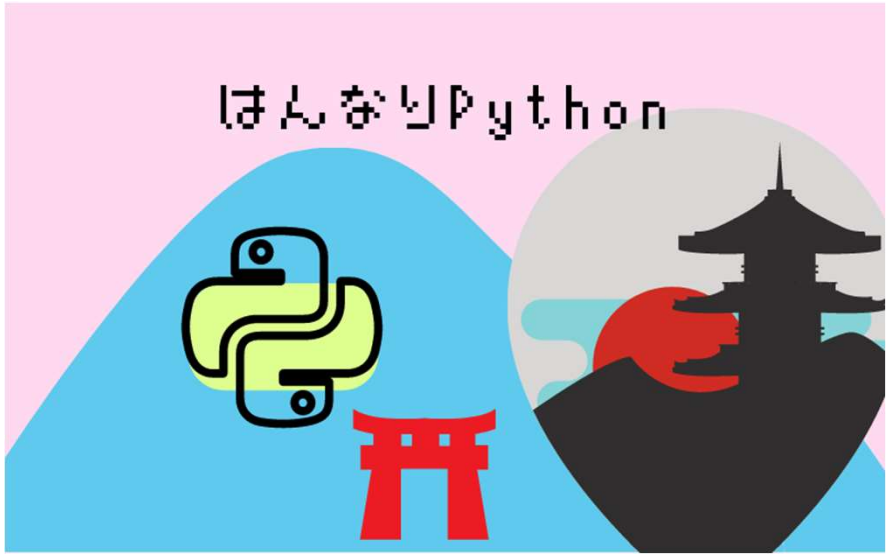
Thank you! PyConJP!!!

Writing

I write about Dash. Dash is plotly’s Web Framework.

Hannari Python

- Python community in Kyoto
- Recently held Online!!!
- Once a week
- Social Tipping!!
- Donate
  - PyCharity
  - Red Cross
  - Japanese Student Service Organization
- https://hannari-python.connpass.com/



I appreciate before my talk.

To the medical community and all those who support society

Thank you!

Motivation for the talk.

- I dealt with a lot of location data at work.
- I realized the importance of location data in dealing with the real world we live in.
- On the other hand, the usage is rarely talked about (me too. according to me).
- Expectation that by talking, other people will start to talk about various things.
- I'll talk about the basics and how to use them.
- Also, there may be some mistakes, so if you notice any, please point them out to me!

Main Packages

- shapely
  - Handling geospatial information
- geopandas
  - Handling geospatial information as tabular data
- xarray
  - Handling multidimensional data
- folium / plotly / pydeck
  - Visualization of geospatial information

agenda

- Location data / Geospatial information
- File Formats
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

agenda

- **Location data / Geospatial information**
- File Formats
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

2020/2 ～ 2021/10

- I can't go out.
- In 2019, I'd be at PyConJP in Tokyo in September, and PyCon China Beijing in October.
  - But these days….
- My area of activity（WFH : 2020/3 - 2021/10 99%）

North: MK BOWL KAMIGAMO

SOUTH: KYOTO STATION (KYOTO TOWER)

WEST: KINKAKUJI

EAST: GINKAKUJI

Use Google Map

- Only people in Kyoto would understand, so let's take a look on Google Maps!
  - Googlemap: My Place
  - https://goo.gl/maps/cFh66Zmg4j7cgfsG7
    google is smart
- Google Maps plots the location by name.
    ex. Ginkakuji
  - Address: 〒606-8402 京都府京都市左京区銀閣寺町2
  - Maybe it's plotted as place name -> address -> coordinates.
  - Search by GinkakujiCoordinates -> Latitude and Longitude: 35.0270° N, 135.7983° E

GeoCoding

- Don't pass addresses when plotting location data.
- Specify by coordinates such as latitude and longitude.
- Conversion from place names and addresses to latitude and longitude is called geocoding.
- Geocoding example (google's geocoding API: API_KEY required)

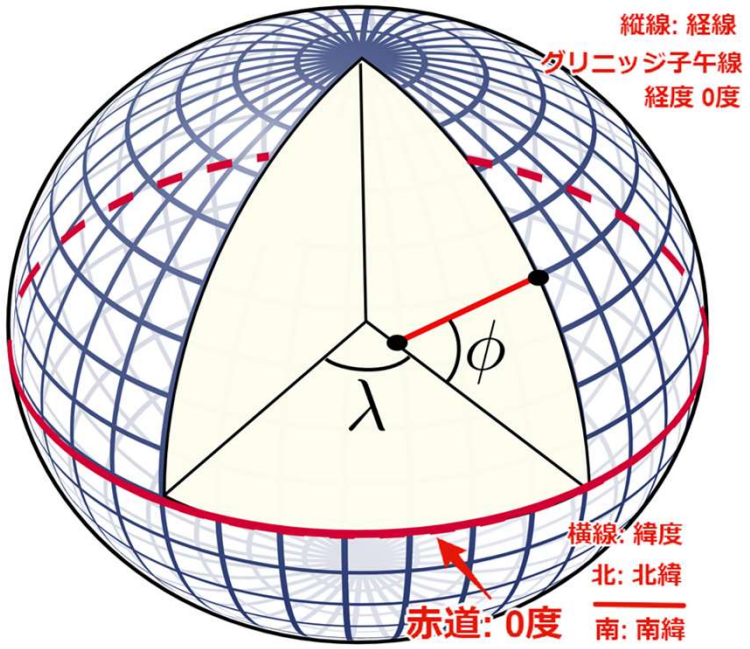
```
[8]   1 import requests

[29]  1 url = 'https://maps.googleapis.com/maps/api/geocode/json'
      2 params = {'address': '銀閣寺', 'key': id_key}
      3 r = requests.get(url, params=params)
      4 result = r.json()['results']
      5 result[0]['geometry']

     {'location': {'lat': 35.0270213, 'lng': 135.7982058},
      'location_type': 'ROOFTOP',
      'viewport': {'northeast': {'lat': 35.0283702802915, 'lng': 135.7995547802915},
       'southwest': {'lat': 35.02567231970851, 'lng': 135.7968568197085}}}
```

Coordinates

- latitude and longitude
  - latitude
    - Represents north and south (up to 90 degrees)
    - 0 degrees from the equator
  - longitude
    - Representation of east and west (up to 180 degrees)
    - 0° prime meridian: 102.478m east of
    - Greenwich Observatory



- Sometimes expressed as decimals, but often expressed in degrees, minutes, and seconds, and the minutes and seconds must be divided by 60.
- Expressing latitude and longitude as numerical values can be used to identify locations.
- On the other hand, there are multiple expressions, and it is very difficult to

image: Peter Mercator, Public domain, from wikimedia commons

Coordinate Reference System（CRS）

- There are multiple definitions for expressing position.
  - A geodesic system that defines what to do with a bumpy elliptical earth.
  - Representing the Earth as a Sphere? Coordinate system for planar representation.
  - A combination of many geodetic and coordinate systems => coordinate reference system
  - According to the GeoRepository site, there are 6547 CRSs.
    - It is run by The International Association of Oil & Gas Producers (IOGP).
  - If you don't pay attention to this, you won't be able to make use of the location data at all.
- Please refer to books and other sources for accurate information.

Shapely

- Use shapely to represent geospatial information
  - https://shapely.readthedocs.io/en/stable/
- Express space using Point, LineString, and Polygon
  - Pass numerical values in the order of x, y (longitude / latitude)
  - There is also Multi * that puts each together.
    - Point : MultiPoint
    - LineString : MultiString
    - Polygon : MultiPolygon
- Notebook Link
  - Colab: https://colab.research.google.com/drive/1CU8jzoLs8Hv0NNmHqES3gtYqneX_PvNJ?usp=sharing

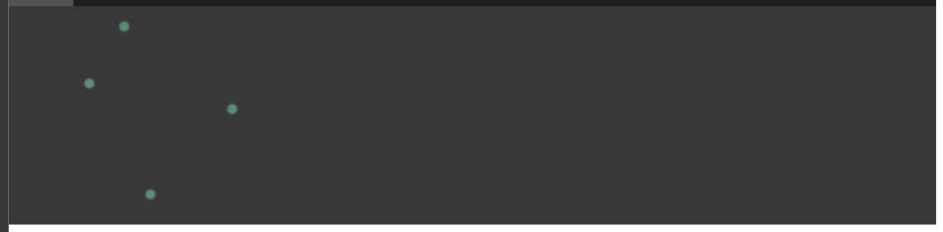
Express the area of my activity with Point

- Expressed in latitude and longitude
- When set to MultiPoint, it is displayed with 4 points

```
[29] 1 df
```

	地名	geometry
0	銀閣寺	POINT (135.79821 35.02702)
1	金閣寺	POINT (135.72924 35.03937)
2	MKボウル	POINT (135.74609 35.06682)
3	京都駅	POINT (135.75877 34.98585)

```
1 center = shapely.geometry.MultiPoint(df.geometry.values)
2 center
```



Express the area of my activity with LineString

- If you pass a Point to multiple LineStrings, it will be represented by a line.
- Let's make a line between east and west, north and south.

```
[32]  1 touzai = shapely.geometry.LineString([df.loc[0, 'geometry'], df.loc[1, 'geometry']])
      2 nanboku = shapely.geometry.LineString([df.loc[2, 'geometry'], df.loc[3, 'geometry']])

[35]  1 print(touzai, nanboku)

LINESTRING (135.7982058 35.0270213, 135.7292431 35.03937) LINESTRING (135.746086 35.0668248, 135.7587667 34.985849)

[34]  1 shapely.geometry.MultiLineString([touzai, nanboku])
```

Express the area of my activity with Polygon

- When multiple Points are passed, they are represented as Polygon.
- Convert four points to a Polygon.

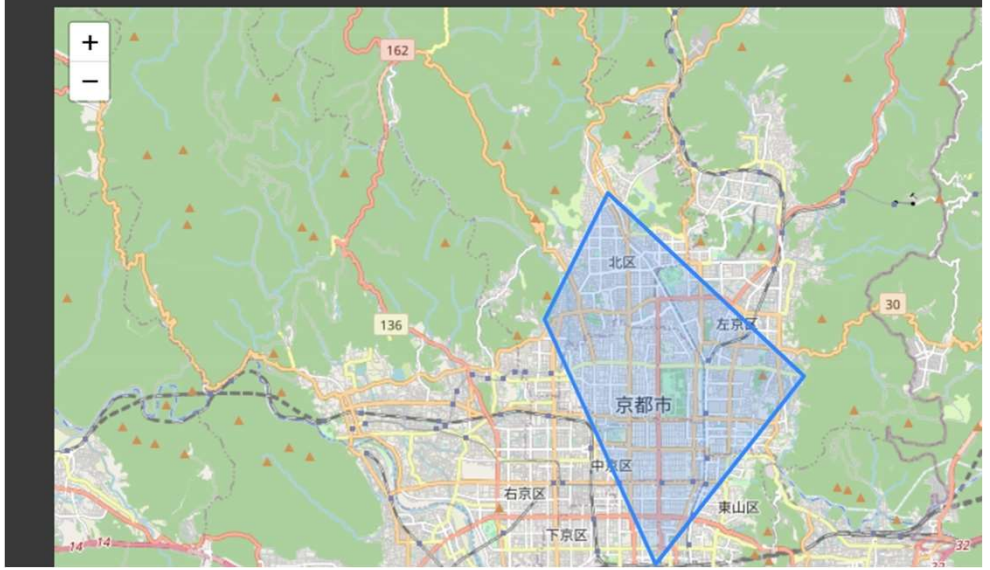
```
[57]  1 polygon = shapely.geometry.Polygon([
      2                                     df.loc[0, 'geometry'],
      3                                     df.loc[2, 'geometry'],
      4                                     df.loc[1, 'geometry'],
      5                                     df.loc[3, 'geometry']
      6                                     ])
```

```
[58]  1 polygon
```

Plotting

- Plotting with folium.

```
[77]  1 polygon_kyoto = shapely.geometry.Polygon([
      2                                 df.loc[0, 'geometry'],
      3                                 df.loc[2, 'geometry'],
      4                                 df.loc[1, 'geometry'],
      5                                 df.loc[3, 'geometry']
      6 ])
      7 center = polygon_kyoto.centroid
      8 map = folium.Map([center.y, center.x], zoom_start=12)
      9 folium.GeoJson(polygon_kyoto).add_to(map)
     10 map
```



Measure distances by transforming the coordinate reference system.

- In this case, EPSG4326 => EPSG6674
- By doing so, we can measure the distance between line segments.
- There are various attributes.

```
[59]  1 df_kyoto = df.to_crs("EPSG:6674")
```

```
[60]  1 df_kyoto
```

	地名	geometry
0	銀閣寺	POINT (-18413.537 -107922.597)
1	金閣寺	POINT (-24702.618 -106537.855)
2	MKボウル	POINT (-23158.194 -103496.311)
3	京都駅	POINT (-22023.357 -112481.852)

```
[70]  1 # 銀閣寺 - 金閣寺の距離
      2 shapely.geometry.LineString([df_kyoto.loc[0, 'geometry'],
      3                              df_kyoto.loc[1, 'geometry']
      4                             ]).length
```

```
6439.7249422724335
```

```
      1 # 京都駅 - MKボウルの距離
      2 shapely.geometry.LineString([df_kyoto.loc[2, 'geometry'],
      3                              df_kyoto.loc[3, 'geometry']
      4                             ]).length
```

```
9056.920195253735
```

agenda

- Location data / Geospatial information
- File Formats
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

agenda

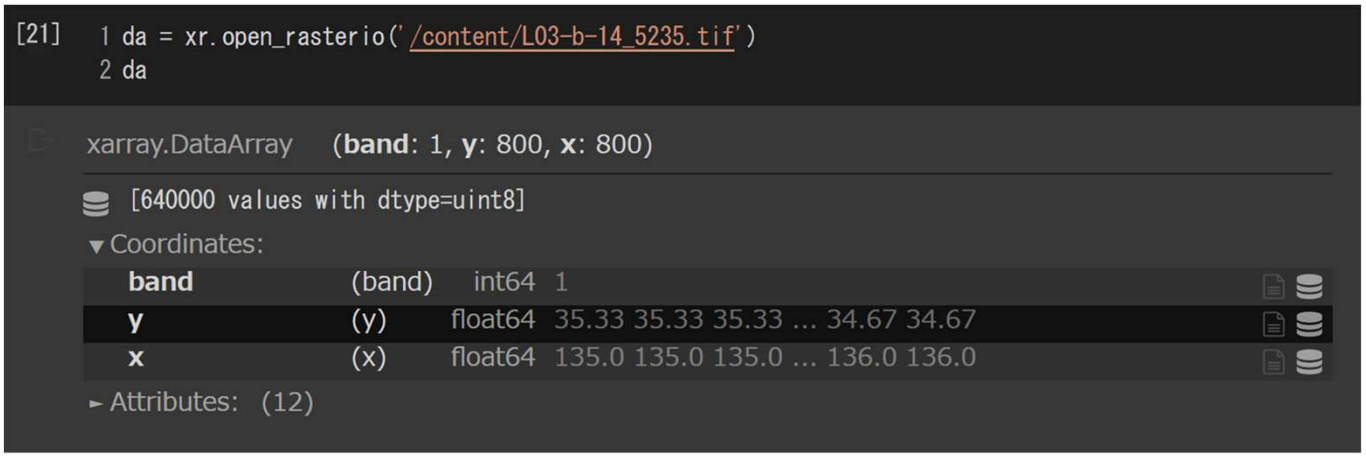
- Location data / Geospatial information
- **File Formats**
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

File Formats

- Raster Data
  - Data is stored in a grid.
  - File format: GeoTIFF, etc.
  - Packages: rasterio, **xarray**
- Vector Data
  - Data created from points, line strings, and polygons
  - File formats: Shape, GeoJSON, KML, etc.
    - Shapefile consists of about 3-5 files.
  - Packages: **geopandas**

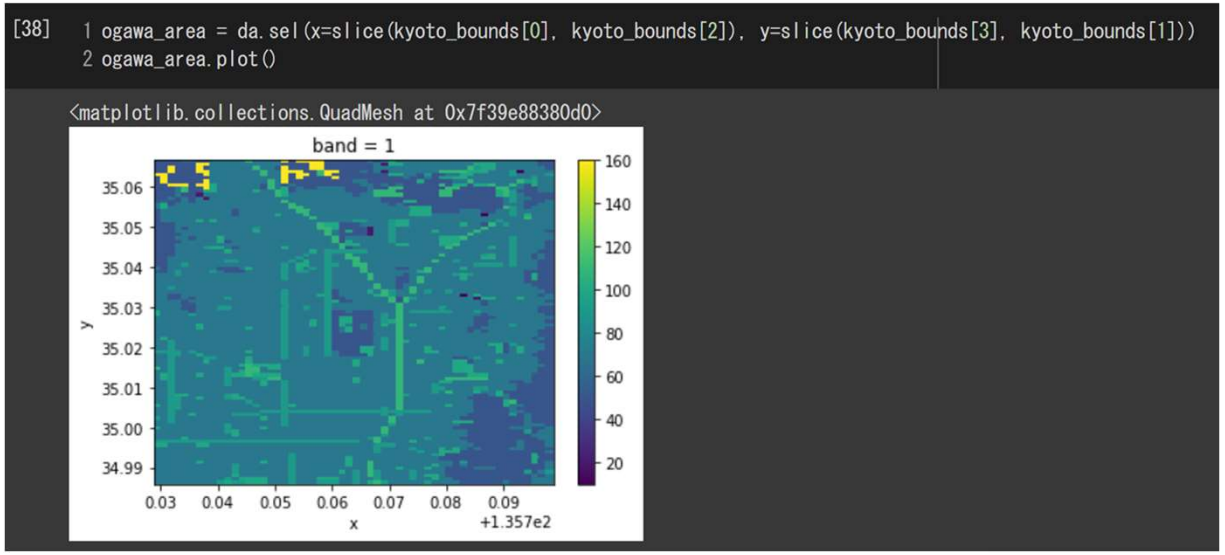
Raster Data

- Data: National Land Information: Land use fine mesh (raster version)
  - https://nlftp.mlit.go.jp/ksj/gml/datalist/KsjTmplt-L03-b_r.html
  - Handling the Kyoto area: 5235
  - Unzip the zip and load the tif file with xarray's open_rasterio function.
  - The data are stored in 100m mesh at each location with the values of land cover classification.
  - Actual reading and objects.



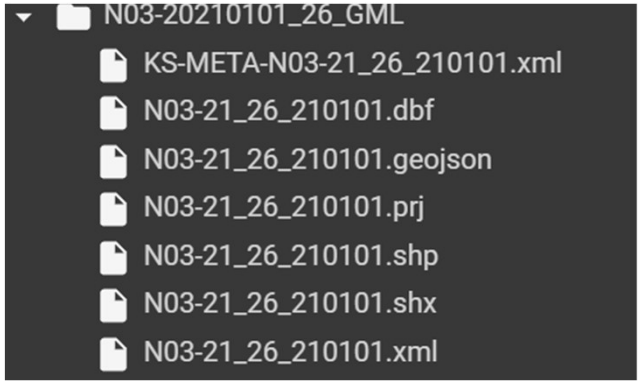
Raster Data 2

- Cut out area of my activity.
  - Can be visualized with the plot method.
  - Using xarray makes it easy to cut out data.



Vector Data

- Data used: National Land Data: Administrative Area Data
  - https://nlftp.mlit.go.jp/ksj/gml/datalist/KsjTmplt-N03-v3_0.html#!
  - As usual, kyoto data.
  - The ZIP contains both shape and geojson.




```
1 data = gpd.read_file('/content/N03-20210101_26_GML/N03-21_26_210101.shp')
2 data.head()
```

	N03_001	N03_002	N03_003	N03_004	N03_007	geometry
0	京都府	None	京都市	北区	26101	POLYGON ((135.72539 35.17080, 135.72552 35.170...
1	京都府	None	京都市	上京区	26102	POLYGON ((135.75062 35.03822, 135.75079 35.038...
2	京都府	None	京都市	左京区	26103	POLYGON ((135.80481 35.31708, 135.80586 35.316...
3	京都府	None	京都市	中京区	26104	POLYGON ((135.73195 35.02251, 135.73195 35.022...

Vector Data 2

- Find out which wards are in my activity.
  - Use intersects method.
  - visualize with folium
  - The light blue area is vector data.
  - With google satellite data.
  - I was surprised by the length of Sakyo Ward.

```
1 data['me'] = data['geometry'].map(lambda x: x.intersects(kyoto_poly))
2 my_data = data[data['me'] == True]
3 my_data
```

	N03_001	N03_002	N03_003	N03_004	N03_007	geometry	me
0	京都府	None	京都市	北区	26101	POLYGON ((135.72539 35.17080, 135.72552 35.170...	True
1	京都府	None	京都市	上京区	26102	POLYGON ((135.75062 35.03822, 135.75079 35.038...	True
2	京都府	None	京都市	左京区	26103	POLYGON ((135.80481 35.31708, 135.80586 35.316...	True
3	京都府	None	京都市	中京区	26104	POLYGON ((135.73195 35.02251, 135.73195 35.022...	True
4	京都府	None	京都市	東山区	26105	POLYGON ((135.78466 35.01035, 135.78468 35.009...	True
5	京都府	None	京都市	下京区	26106	POLYGON ((135.77017 35.00428, 135.77019 35.003...	True

agenda

- Location data / Geospatial information
- File Formats
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

agenda

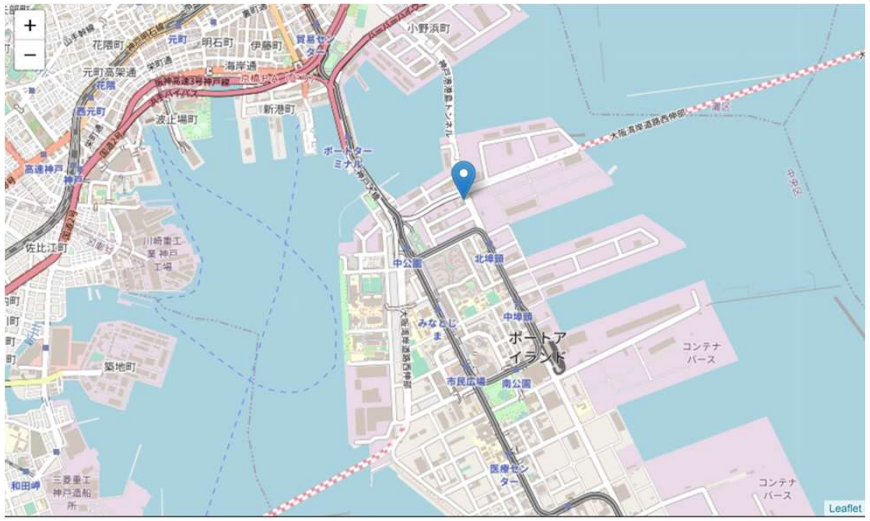
- Location data / Geospatial information
- File Formats
- **How to use**
  - **Personal behavior data**
  - National census
  - Himawari
  - Satellite Images

Personal behavior data

- Data: GPS trajectory linked data project
  - https://github.com/koujikozaki/GPS2LOD (CC4.0)
  - CSV file
  - Description: Behavior during a conference
  - We have time and location data, altitude, speed, etc.
  - I'll observe the data to see if everyone was acting seriously.
  - colab: https://colab.research.google.com/drive/1_LNM-AKdpcuJb-vOzPKJt3O81o1jOGia?usp=sharing
- data processing
  - We will use speed, location, and personal information.
  - Converted the data to every 30 minutes.
  - Point type for position information that exists as a float.
  - DataFrame => GeoDataFrame
  - CSV => GeoJson

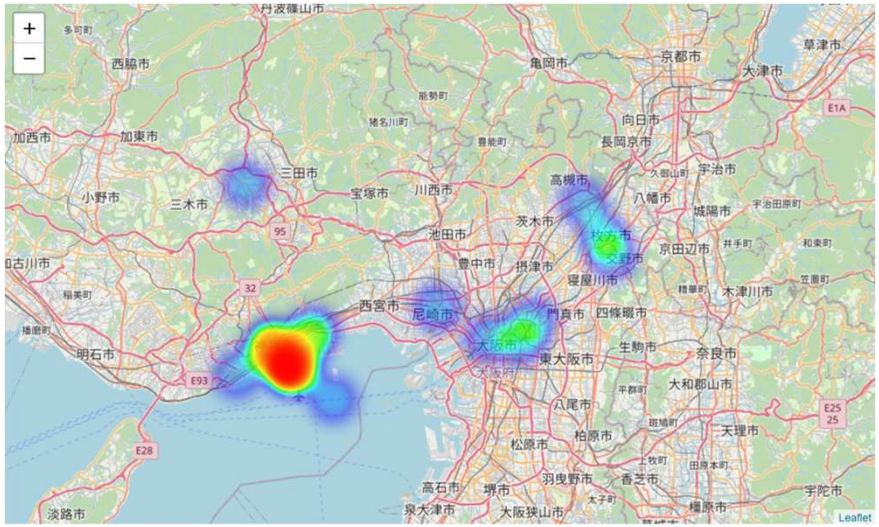
Personal behavior data 2

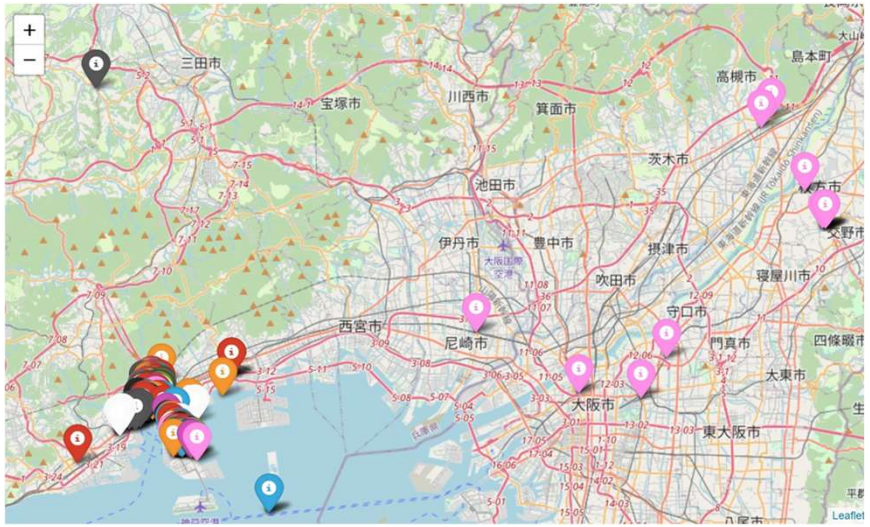
- What is the central point of everyone's actions?
  - It can be obtained with the centroid attribute of MultiPoint, but it is the center of movement... (left)
  - I created a median from position data that is a float (right): Bingo!



Personal behavior data 3

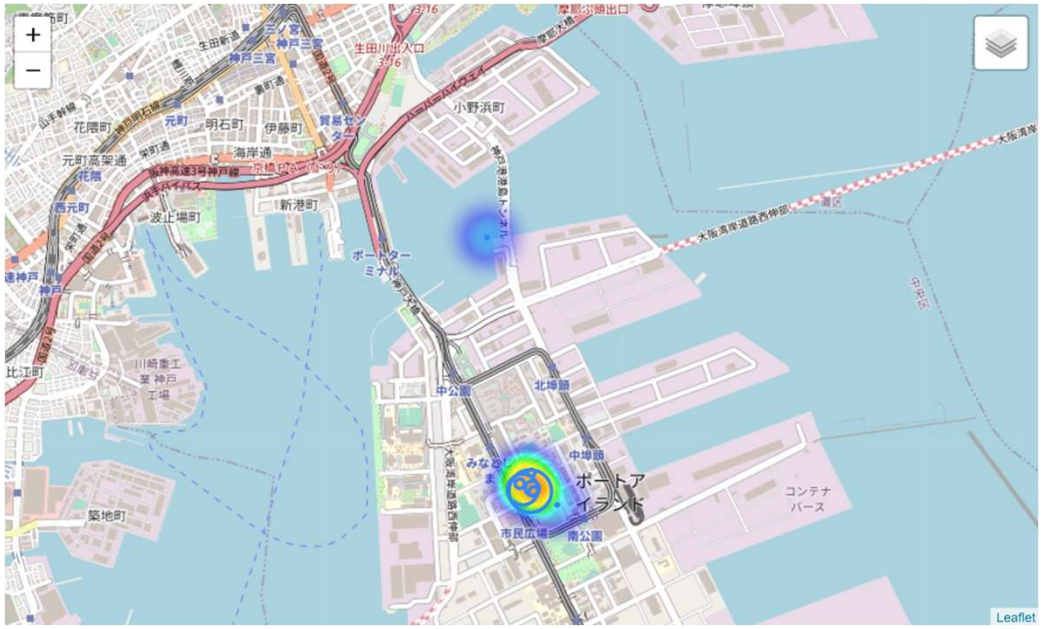
- Observe the behavior of all people.
- Use heat maps to observe where the frequency was highest.
- Observe each individual by separating them by color.





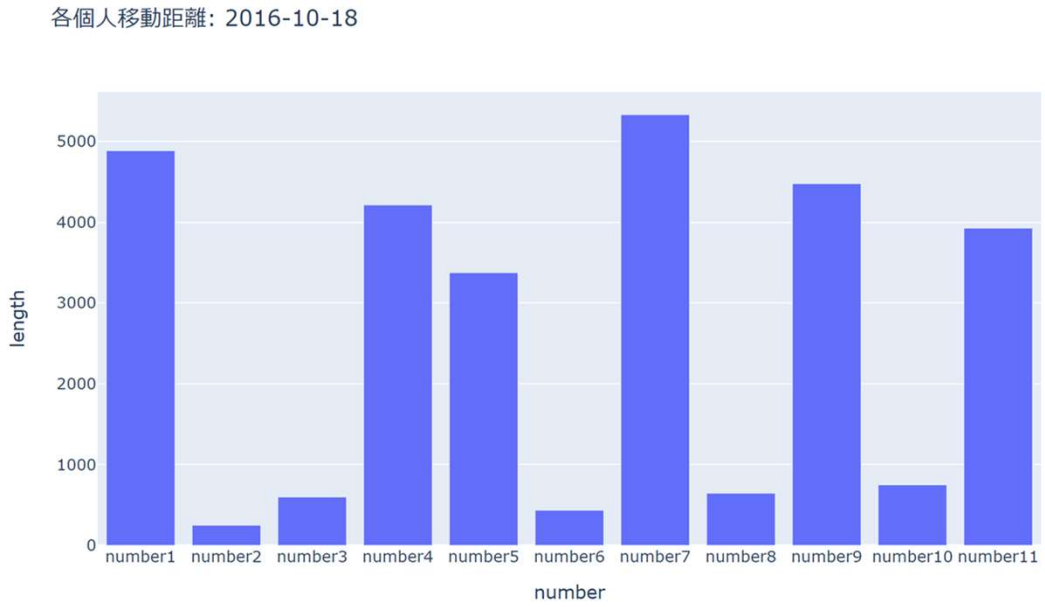
Personal behavior data 4

- The speed of each individual at a given point in time is represented by the size of the circle.
- Heat map representation of how many people are gathering.



Personal behavior data 4

- Compare the daily latitude distance for each individual
- Change the coordinate reference system to express in meters.（EPSG4326 => EPSG6673）



agenda

- Location data / Geospatial information
- File Formats
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

agenda

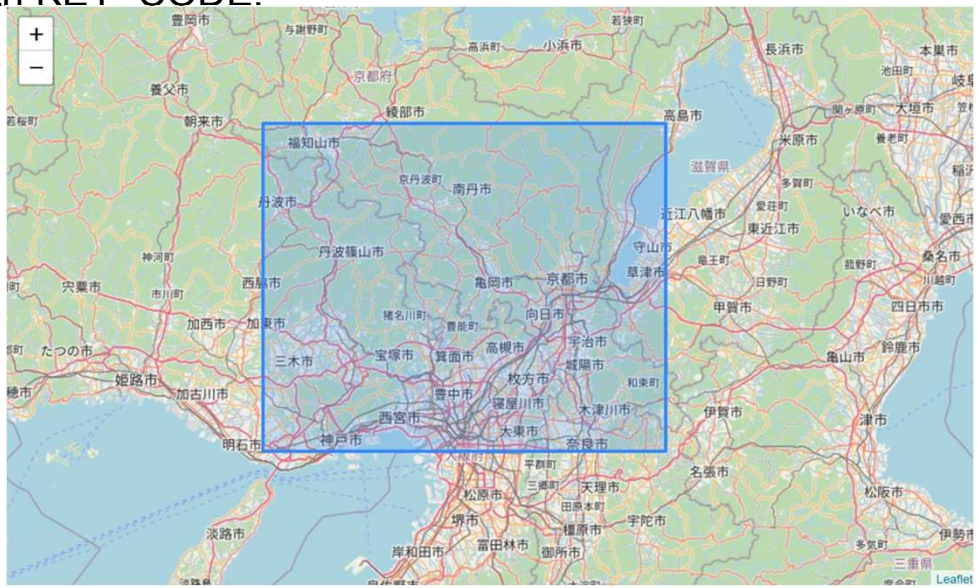
- Location data / Geospatial information
- File Formats
- **How to use**
  - Personal behavior data
  - **National census**
  - Himawari
  - Satellite Images

National census

- Census is the most important statistical survey of the country.
  - We can also get data on the number of people in each household, their ages, etc.
  - Nationwide, divided by town/character, data created by 1km, 500m, 250m mesh.
  - Statistical data and boundary data are kept separately, and regional data is created by merging the two.
  - Can be obtained in csv and shape file(zipped up).
  - https://www.e-stat.go.jp/gis
- In this article, we will show examples of marketing using basic statistics such as population.
  - Marketing to ages 0-14
  - Marketing to foreign residents
- Colab: https://colab.research.google.com/drive/1QspBpoW9BO_ofXmrLOX3a04YD5SE_3eH?usp=sharing

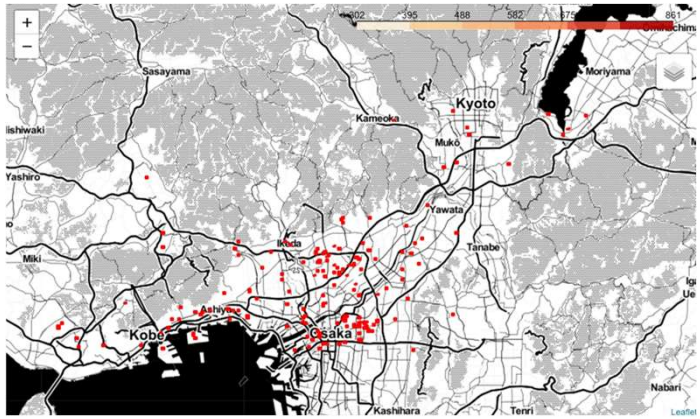
National census 2

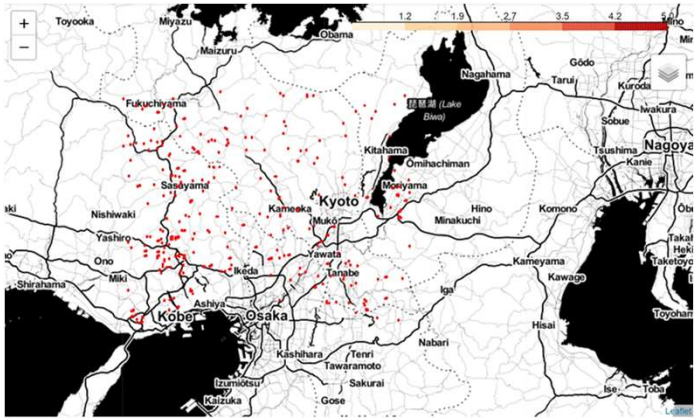
- We have created a data set that includes the central part of Kyoto.
  - Data on total population, sex ratio, age structure, number of households, etc. on a 250-meter mesh
  - Merged with KEY_CODE.



National census 3

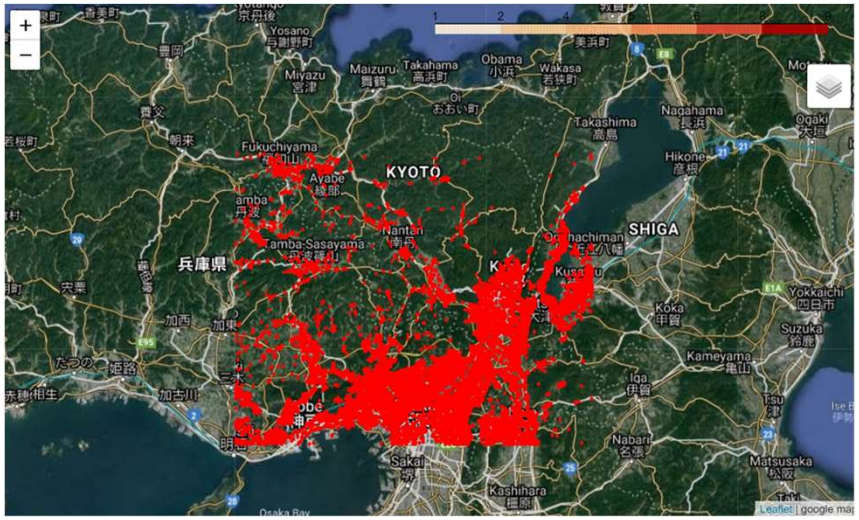
- Use the 0-14 year old population to find areas that are likely to have a younger population.
  - Separate by total number and ratio (300+, 40%)
  - This area can be created with Pandas-like processing.





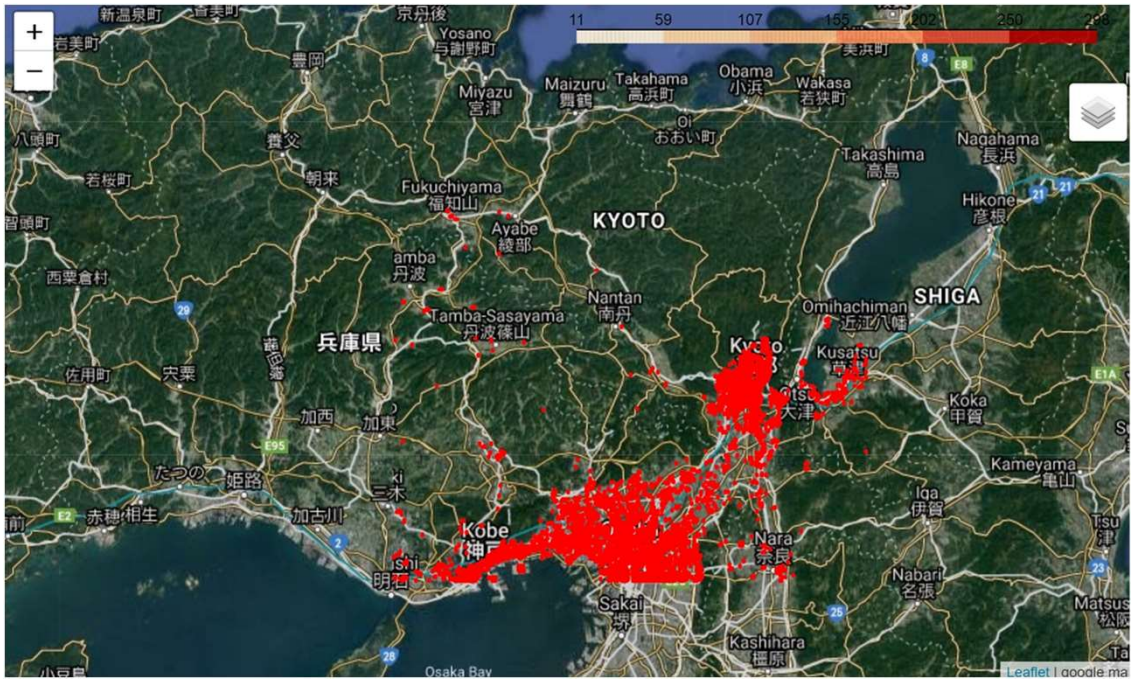
National census 4

- Delivering information to foreign residents.
- The less foreigners they have nearby, the more difficult it will be to get information.
- Find areas with less than 10 people on a 250 meter mesh.



National census 5

- There were more people under 10 than I expected, so I checked for people over 10.



agenda

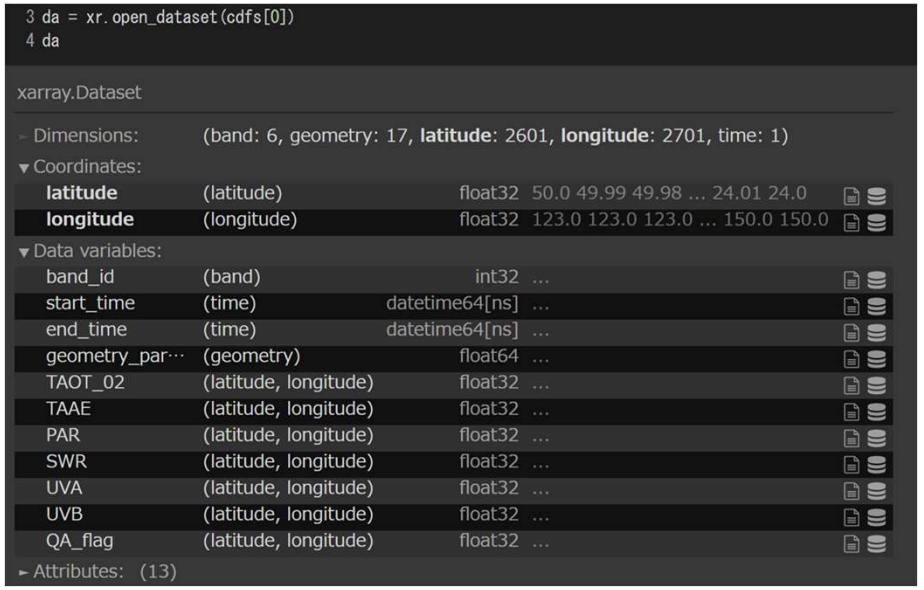
- Location data / Geospatial information
- File Formats
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

agenda

- Location data / Geospatial information
- File Formats
- **How to use**
  - Personal behavior data
  - National census
  - **Himawari**
  - Satellite Images

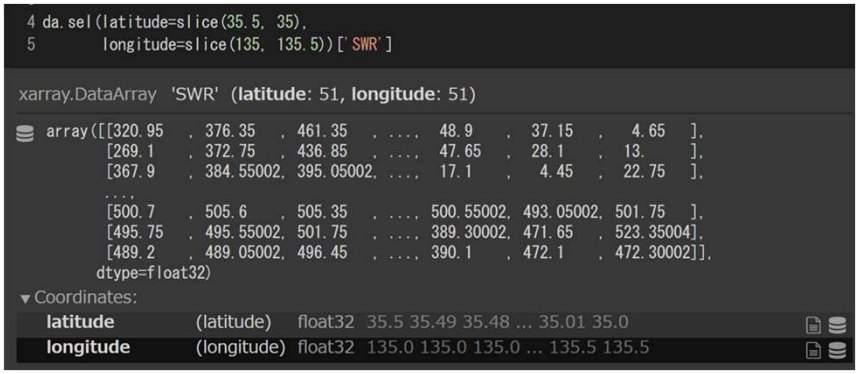
Meteorological data with xarray

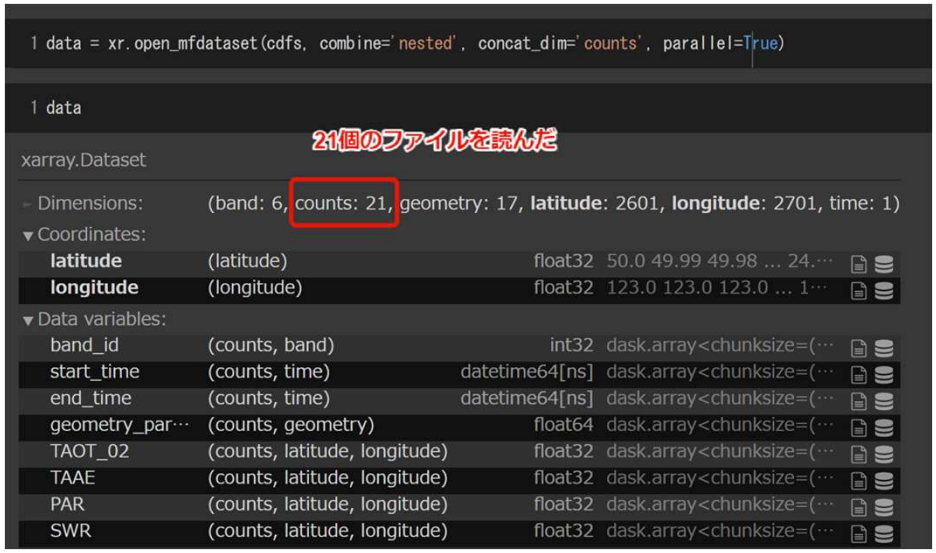
- Data: PTree 「(c) JAXA・気象庁」
  - Colab: https://colab.research.google.com/drive/1J3QU3Hv2iO1EKYGKma96CkkFeVqqzzrD?usp=sharing
- Description https://www.eorc.jaxa.jp/ptree/userguide_j.html
  - Non-commercial use only.
  - Buy it if it's commercial use!
- Data is in NetCDF format
  - Multidimensional Array
  - Use xarray
- What to do
  - See how the amount of sunlight
  - changes over the of a day.



Meteorological data with xarray 2

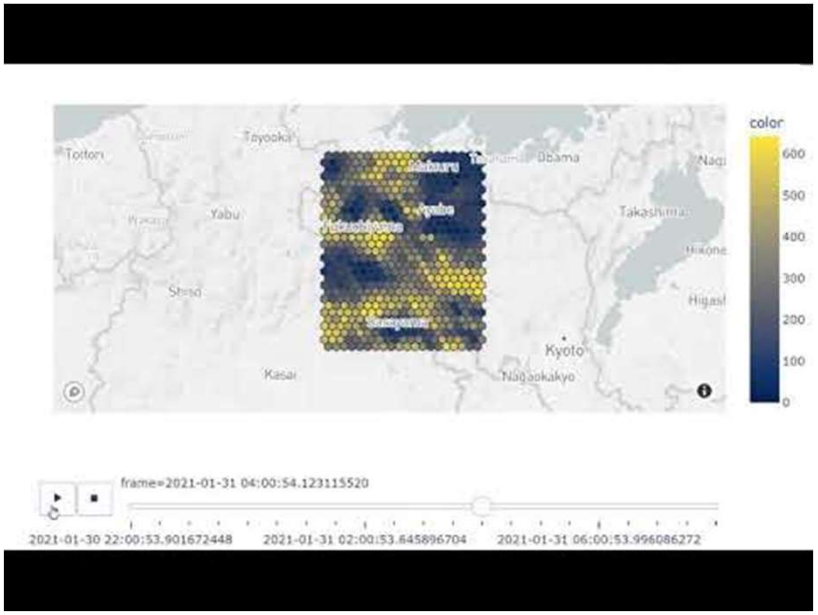
- Using xarray is intuitive and easy to understand.
  - I think I can get there if I can use pandas.
  - Files can be readed together.





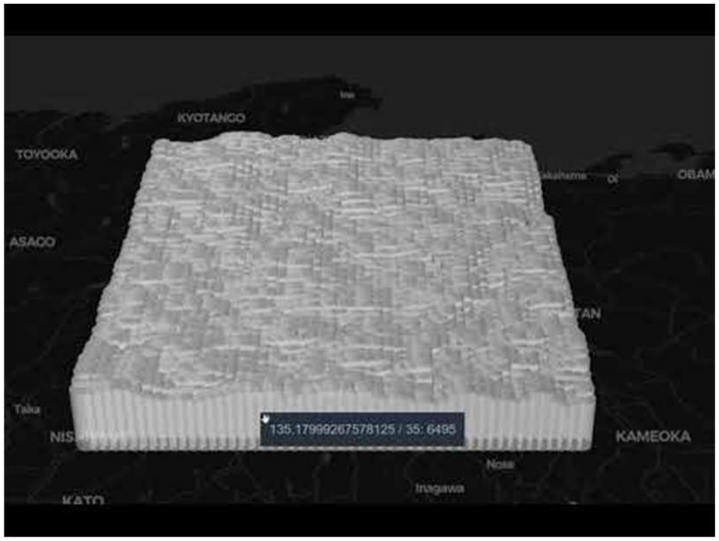
Meteorological data with xarray and plotly

- Using Plotly to visualize hourly data.
  - The more graph move, the more you feel!



Meteorological data with xarray and pydeck

- Express the total for the day.
  - Group by location and total for one day.
  - With pydeck.



agenda

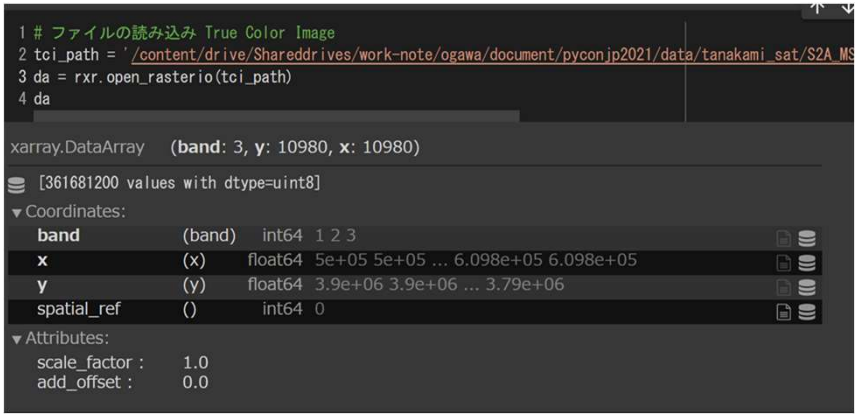
- Location data / Geospatial information
- File Formats
- How to use
  - Personal behavior data
  - National census
  - Himawari
  - Satellite Images

agenda

- Location data / Geospatial information
- File Formats
- **How to use**
  - Personal behavior data
  - National census
  - Himawari
  - **Satellite Images**

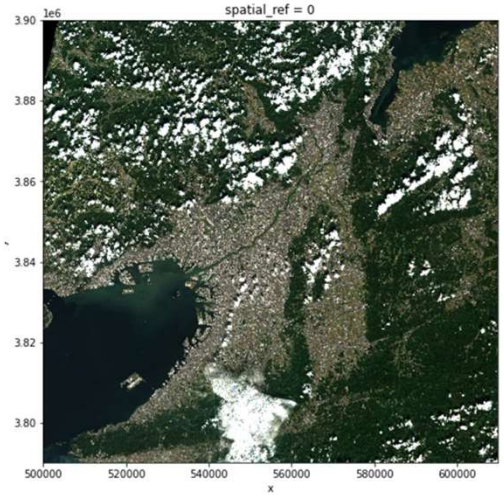
Satellite Images

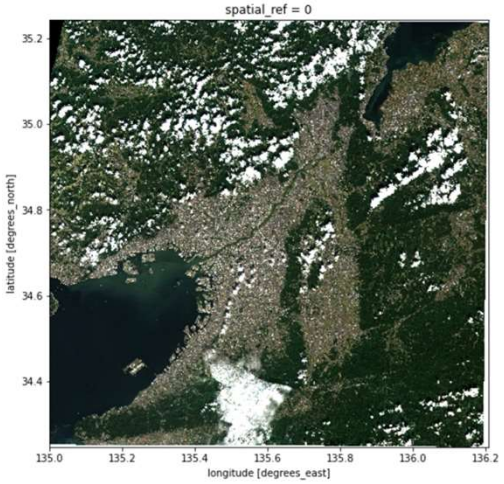
- Data: Sentinel2
  - Colab: https://colab.research.google.com/drive/1hajaEIBbUwGWvm5JbnJ6y0WhsMDwc3Yi?usp=sharing
  - License Copernicus Open Access Hub (Link)
  - There are various files, but the raster data is in a jp2 file.
  - Read and process the file using rioxarray.
  - Read 1pixel / 10m TCI image
  - Observe the local Mt. Tanakami (I can only go back a year for convenience, so I will compare the two)



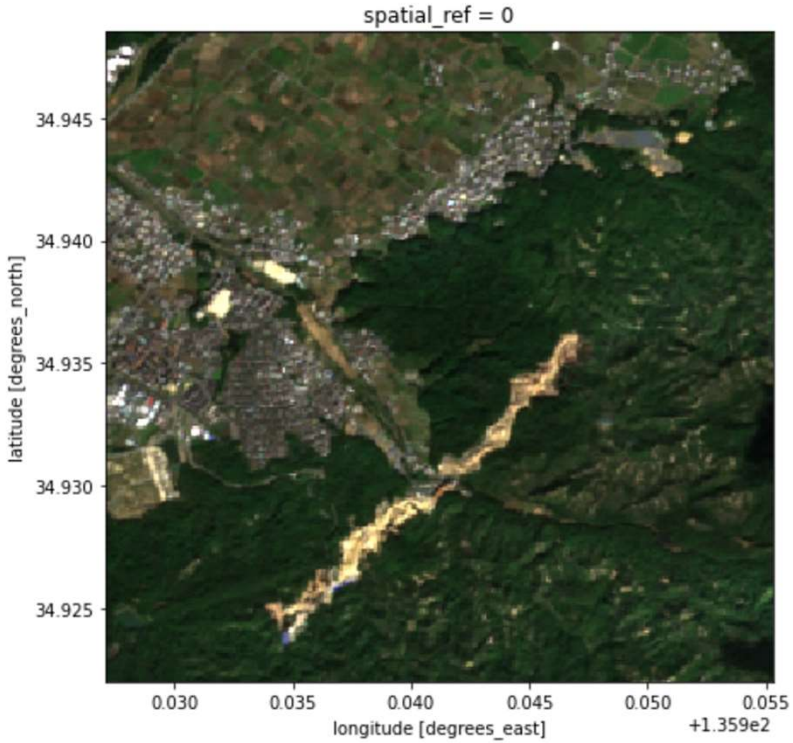
Satellite Images 2

- Visualization
  - Satellite images of Osaka, Kyoto, Nara, and Shiga
  - Changing the CRS（EPSG:32653 => EPSG:4326）
  - Call up the location data of the part of Mt. Tanakami you want to see that you prepared in advance, and cut it out.

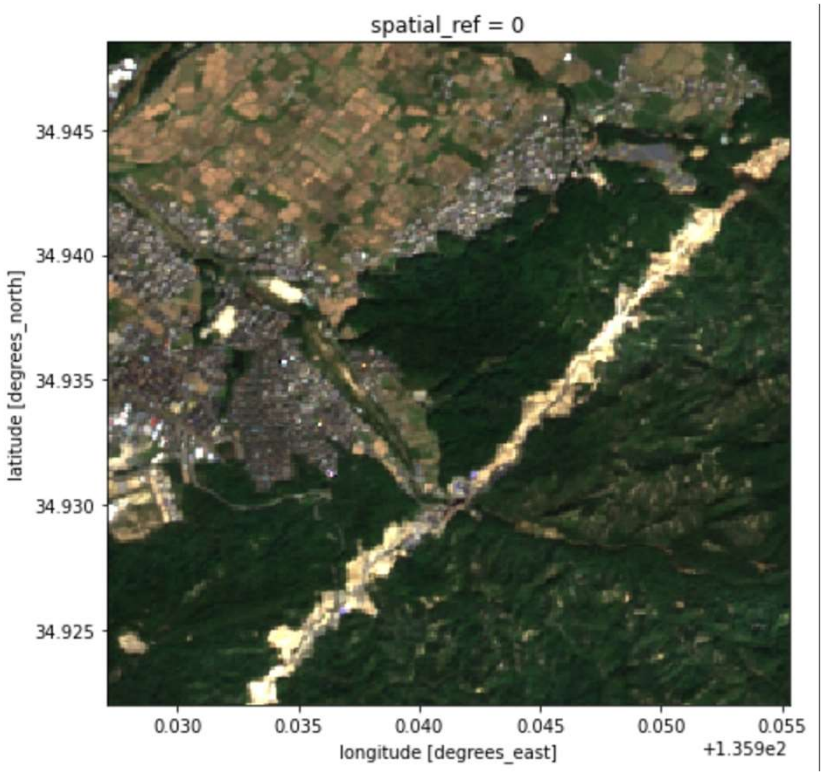




Satellite Images 3 : 2020/10/12



Satellite Images 4 : 2021/10/2



conclusion

- It's easy to make use of location data in Python.
- On the other hand, the exact knowledge is quite complex, and I'd like to work with an expert to touch this area.
- Once the data is in place, it will be possible to help solve social issues.
- "How do we act?" may be the challenge for those of us who can touch data or write code.

conclusion 2

from ghibli
https://www.ghibli.jp/works/mononoke/#frame

Thank you very much

参考資料

books(only japanese)

- 地図リテラシー入門　羽田康祐　ペレ出版
- その問題、デジタル地図が解決します　中島円　ペレ出版
- GIS地理情報システム　矢野桂司　創元社

for study

- GEO-PYTHON(University of Helsinki) Link

Packeges

- shapely
  - Document: https://shapely.readthedocs.io/en/stable/
- geopandas
  - Document: https://geopandas.org/
- xarray
  - Document: http://xarray.pydata.org/en/stable/
- folium
  - Document: https://python-visualization.github.io/folium/
- plotly
  - Document: https://plotly.com/python/
- pydeck
  - Document: https://deckgl.readthedocs.io/en/latest/

Data

- Behavior data of 11 people
  - GPS trajectory linked data project Link
- kokuse-chosa
  - eStat Statistics GIS https://www.e-stat.go.jp/gis
  - kokuse-chosa (2015) 250meter mesh
- Himawari
  - JAXA HIMAWARI MONITOR: Link
- Sentinel2
  - Copernicus Open Access Hub: Link