Code documentation

Dataset

The dataset is from Amazon web services, the link for the dataset is https://registry.opendata.aws/usgs-lidar/. Here we have to find a way to fetch the data and use it for processing.

A python class is used for fetching data from AWS using **boto3,** the AWS SDK for Python. Boto3 makes it easy to integrate our Python application, library, or script with AWS services including Amazon S3, Amazon EC2, Amazon DynamoDB, and more.

Organizing, storing and retrieving data in Amazon S3 focuses on two main things — buckets and objects(file) that work together to create your storage system.

In the code api.py I used **usgs-lidar-public** as a bucket and an **AK_BrooksCamp_2012** file or object. And running the code downloaded the **AK_BrooksCamp_2012** file for processing. You can find other files in the filename.txt .And it shows that we can retrieve any data from the AWS and also download files using python code and boto3 SDK.The file AK_BrooksCamp_2012 contains las, json and js files. And the total size is around 2.1GB.

Coordinate system

There are numerous conventions used globally for representing the coordinate system for map data

- PROJ4- http://projg.org/usage/projections.html
- OGC WKT(de facto standard)
- EPSG codes(easy to use)

- XML etc

Data Fetching and Loading

The api.py code contains a python code to fetch data from aws just by importing boto3 sdk and using bucket name and file name.

The api.py gives us a list of LAZ files and converts it to a raster file and accesses it using GDAL which is a library. It presents a single raster abstract data model and single vector abstract data model to the calling application for all supported formats. It also comes with a variety of useful command line utilities for data translation and processing.

Also loading the single raster files width and height , coordinate system and description, gives us the shape of the raster file and its coordinate system information.

GeoTransform contains the coordinates (in some projection) of the upper left (UL) corner of the image (taken to be the **borders of the pixel** in the UL corner, not the center), the pixel spacing and an additional rotation.

Getting the number of raster bands each band is a data set; moreover, the raster data set may contain sub-data sets, and each sub-data set may contain bands. The number of bands here tells us that it's a single band raster i.e either binary image represented as 0 or 1 or grayscale image represented from 0 to 255 or color map.

We will read in the raster data as a numerical array in order to use the capabilities in the NumPy-library. Converting existing Gdal Dataset or a Band into a numpy array

Data visualization

Visualizing the raster array data by plotting it into different formats and also using the rasterio package.

Python code for submitting a boundary (GPS coordinates polygon) and receiving back a raster of the height of the terrain within the boundary. The expected inputs and outputs are

Inputs:

- Field boundary polygon in geopandas dataframe
    - All CRS's (coordinate reference systems) should be accepted
- Desired output CRS

Outputs:

- **Python dictionary** with all years of data available and geopandas grid point file with elevations encoded in the requested CRS.

Terrain Visualization

A code that graphically displays the returned elevation files as either a 3D render plot or as a heatmap.