Exercise: Building a Buffer Around a Highway

Summary

This exercise uses geopandas to project Census shapefiles appropriately for New York State. It then builds several map layers for Onondaga County, including a 1-km buffer around the county's interstate highways.

Input Data

There are two input files, both of which are TIGER/Line shapefiles that will be downloaded from the Census in part A of the instructions.

Deliverables

There is one deliverables: a script called **onondaga.py** that builds a geopackage and draws a figure. The geopackage file and figure will be reconstructed by running your script and won't be uploaded to GitHub.

Instructions

A. Downloading files from the Census

1. Go to the Census TIGER/Line page (there's a link on the class web site) and then click on the link for the web interface.
2. Select 2020 and then download two files: "Counties (and equivalent)", which should be `tl_2020_us_county.zip`, and "Roads > Primary and Secondary Roads > New York", which should be `tl_2020_36_prisecroads.zip`.

B. Script onondaga.py

1. Import `geopandas` as `gpd`.
2. import `matplotlib.pyplot` as `plt`.
3. Import `os`, a module that gives access to a range of operating system functions.
4. Set variable `utm18n` equal to 26918, which is the EPSG code for UTM 18N using the NAD83 coordinate system. It's the projection recommended by the NYS GIS Clearinghouse and usually used by NYS agencies. The units of the projected coordinate system will be meters.
5. Set variable `out_file` to `"onondaga.gpkg"`, which will be a geopackage produced by the script.

   Read the input files and then filter and project them:
6. Use the `gpd.read_file()` function to read the Census county shapefile into a variable called `county`.
7. Create a variable called `on_border` by using the `.query()` method of `county` to select Onondaga County by comparing the `GEOID` field to the county's FIPS code, `"36067"`.
8. Set `on_border` to the result of calling the `.to_crs()` method on `on_border` with the argument `epsg=utm18n`. (FAQ 1)
9. Use the `gpd.read_file()` function to read the primary and secondary roads file into a variable called `roads`.
10. Create a variable called `inter` by using the `.query()` method of `roads` to select the records where the route type, `"RTTYP"`, is equal to `"I"`, the code for interstates.
11. Set `inter` to the result of projecting `inter` to UTM 18N following the approach used for `on_border`.

    Clip the roads at the county boundary:

12. Now clip the interstate layer at the county boundary by setting `on_int_all` to the result of calling the `.clip()` method on `inter` with two arguments: `on_border`, which provides the border for clipping, and `keep_geom_type=True`, which indicates that the clipped file should have the same type of features as those in the layer being clipped (lines in this case, but polygons in other cases). (FAQ 2)

    Dissolve the interstates layer:

13. Now dissolve the features in the interstate layer to create a single feature representing all of the interstates in the county. The dissolved layer should be called `on_int_dis` and it should be created by calling the `.dissolve()` method of `on_int_all` with the arguments `by="RTTYP"` and `aggfunc="first"`. See the Tips section for more information about what dissolving a layer does.

14. Set `on_int_dis` to the result of calling `.reset_index()` on itself, `on_int_dis`. This has the effect of moving the index, which is "RTTYP" after the dissolve, into an ordinary column. That prevents the index information from being lost in subsequent operations that don't preserve it. It's not a big deal here since there's only one value of RTTYP but can be very important in other cases.

    Build a buffer around the interstates:

15. Set variable `radius_m` to 1000. That will be the radius of a buffer in the units of the projection, which is meters in this case.

16. Now create a layer called `buffer` that is equal to the result of calling the `.buffer()` method of `on_int_dis` with argument `radius_m`.

    Create a new geopackage file of the results:

17. Use an if statement to check whether the output file already exists by calling `os.path.exists()` with `out_file` as the argument.

18. If the file does exist, delete it by calling `os.remove()` with argument `out_file`. There's no need for an else statement.

    Save the results in a geopackage:

19. Save `on_border` to a geopackage file by using its `.to_file()` method with `out_file` as the file name and `"county"` as the layer name.

20. Save `on_int_dis` to the same geopackage file but with layer name `"interstates"`.

21. Save `buffer` to the same geopackage file but with layer name `"buffer"`.

    Construct a figure:

22. Create a new single panel figure with figure and axes given by `fig` and `ax1` and using `dpi=300`.

23. Plot `on_border` by calling its `.plot()` method with arguments `color="tan"` and `ax=ax1`.

24. Plot `buffer` by calling its `.plot()` method with arguments `color="tomato"` and `ax=ax1`.

25. Plot `on_int_dis` by calling its `.plot()` method with arguments `color="black"`, `linewidth=0.5`, and `ax=ax1`.

26. Turn off the axis labels by calling `ax1.axis("off")`.

27. Tighten the layout and save the figure as `"highway.png"`.

Submitting

Once you're happy with everything and have committed all of the changes to your local repository, please push the changes to GitHub. At that point, you're done: you have submitted your answer.

Tips

- Dissolving a layer is a form of aggregation and is the geographic equivalent of combining the Pandas `groupby()` and `agg()` functions. The `by=` argument indicates how the groups should be formed: here it says that all features with the same value of `"RTTYP"` should be grouped together (all the interstate segments, in this case). The `aggfunc=` argument indicates how the attributes for the grouped data are to be set. Here, `"first"` says that the attributes should be set to their values for the first object in each group. There are a number of options, including `"first"`, `"last"`, `"sum"`, `"max"`, `"mean"`, and `"median"`. However, we're not going to use the attributes so we'll use `"first"` for simplicity since it works for both string and numeric fields. It's also the default if no option is specified.
- This is the start of a multi-part exercise that will involve classifying residential properties in the county by their proximity to the interstates.

FAQs

1. Be aware that there are two coordinate-related methods with very similar names that do drastically different things. The one to use here is `.to_crs()`: it goes through the geographic data and converts it to the desired projection. The one *not* to use here is `.set_crs()`. That simply sets a variable indicating what projection was used in building the data but doesn't change any of the data. It's used when building a new GeoSeries or GeoDataFrame from scratch.
2. As a general rule, always use `keep_geom_type=True` when clipping in geopandas. The most important thing it does is to close polygons when they are clipped through the middle. Without it, a polygon like that ends up as sequence of line segments representing the part of the polygon's border that happened to be inside the clipping area. With it, a smaller polygon is returned that has been chopped off at the border of the clipping region, which is usually what you want. Clipping in QGIS does this automatically.