Visualising Wildfire Proximity to Populated Areas and Infrastructure in California – Technical Report (How to Guide) by Daniel Stratford B00992246, 6th May 2025.

1. Introduction

Wildfires pose a significant environmental and societal risk in many parts of the world, with California being especially vulnerable due to its climate and vegetation. Data from the California State Geoportal allows us to assess the proximity of wildfires to populated areas and state infrastructure, helping inform evacuation planning and resource allocation.

According to Palinkas (2020), displacement caused by wildfires can be categorised as temporary, long-term, or permanent, depending on the extent of the damage. To better understand these risks, a Python script has been developed to visualise wildfire proximity relative to urban areas and evacuation routes, specifically state highways. The script enables repeatable analysis using data from the California State Geoportal and updated satellite imagery.

The script was developed using the PyCharm IDE and is version-controlled via GitHub, with branching and repository management handled directly through the IDE. Once configured, the script requires no additional branches for successful execution unless a user requires amendments to be made to suit an alternative purpose, a branch has been made available in GitHub for an alternative output.

2. Setup and Installation

Software Requirements

- **Anaconda Navigator:** This is required to upload the environment file and packages check, an account is required.
- **PyCharm Community Edition:** This is required for script development, modification and execution.
- **GitHub:** An account is required for the creation of a repository.
- **GitHub Desktop:** This is optional, for repository management and installation is at user discretion.

Account Requirements

- **USGS Earth Explorer:** Required for downloading satellite imagery (account is necessary for download).

Repository Access

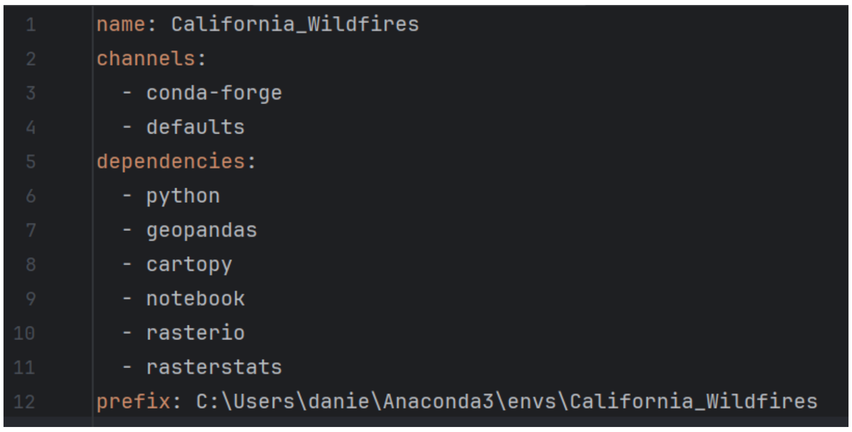
A GitHub repository containing the required files (excluding imagery) is publicly available here GitHub Repository. Forking this repository into your own GitHub account is recommended over cloning to preserve the original while allowing modifications. For more information on how to fork and clone a repository please visit Fork a repository. GitHub provides a back up of the data, files and scripts required to successfully execute the program. It has an easy-to-use API and can be retrieved by accessing a URL through a HTTP connection (Hu et al., 2016).

Working Environment

Create a local working directory where data will be stored. After importing the Californian_Wildfires.yml environment file into Anaconda, a corresponding workspace will be created under the Anaconda3/envs/ directory. This mirrors the folder structure in the local GitHub directory and can also be seen in the GitHub folder within typically C:/Users/<name>/GitHub.

Dependencies

The environment file (Californian_Wildfires.yml) lists all required Python libraries. A visual of the dependencies can be found in *Figure 1 Dependencies*.



```yaml
name: California_Wildfires
channels:
  - conda-forge
  - defaults
dependencies:
  - python
  - geopandas
  - cartopy
  - notebook
  - rasterio
  - rasterstats
prefix: C:\Users\danie\Anaconda3\envs\California_Wildfires
```

Figure 1: Dependencies -Source: yml file found in repository showing the dependencies required for the script.

3. Required Datasets

Vector Data

The following shapefiles, sourced from the California State Geoportal and available via the previously listed GitHub repository where the data has been optimised, are required:

- California County Boundaries.
- State Highways.
- Urban Areas.
- Wildfire Perimeters (Fires_50000).

These datasets are pre-processed and clipped but can be updated with newer data, if available via the sources stated in table 1 and managed through ArcGIS Pro, it is worth noting that data is managed due to GitHub file size limitations.

Raster Data

A Landsat 8-9 imagery tile is required to be downloaded from USGS Earth Explorer.

- **Tile name**: LC09_L2SP_044034_20250409_20250411_02_T1
- **Source**: USGS Earth Explorer
- **Type**: OLI/TIRS C2 Level 2

This tile represents the Area of Interest (AOI) for the analysis and must be referenced within the Python script. To narrow the search down use the state of California as a search area. Below in Table 1: Dataset Summary the data and sources can be seen.

Table 1: Dataset Summary – Source: Original created in MS Word showing datasets used and required for script to work.

Dataset	**Type**	**Source**	**Remarks**
Landsat 8-9 OLI/TIRS C2 L2	Raster MTL	USGS Earth Explorer	Composite image to be created
California County Boundaries	Shp	California County Boundaries	Data managed and located in GitHub repository
State Highway	Shp	California State Highways	Data managed and located in GitHub repository
Urban Areas	Shp	California Adjusted Urban Area	Data managed and located in GitHub repository
Fires_50000	Shp	Wildfire Perimeter	Data managed and located in GitHub repository

4. Data Preparation

Vector Data

Minimal preparation is required; pre-processed shapefiles are included in the repository. If replacing with newer data, use ArcGIS Pro to reproject and clip the data as necessary.

Raster Data

The Landsat image must be converted to a composite image using bands 4 (Red), 3 (Green), and 2 (Blue). This is achieved in ArcGIS Pro and exported as a .tif file. The CRS should be WGS84 UTM Zone 10N which will match the vector datasets and ensure correct grid information in the outputs.

5. Methodology

The process to generate the outputs consists of the following steps:

1. Fork / clone the GitHub repository.
2. Download Landsat imagery from USGS Earth Explorer.

3. Import Californian_Wildfires.yml into Anaconda Navigator to create the working environment.
4. Prepare the imagery and (if needed) vector data in ArcGIS Pro.
5. Open PyCharm and link it to your local GitHub repository.
6. Modify the script to reference correct file paths for imagery and vector data.
7. Customise map extent, symbology, and marginalia as needed.
8. Execute the script and evaluate outputs.

The script outputs both a .png for presentations and a .tif for spatial analysis in GIS platforms or online publication via a portal.

GitHub version control enables collaborative development and secure backup. The gitignore file excludes unnecessary files, and the repository includes a general use license which allows users the freedom to use, share and modify and a README file that describes what the script does and the fundamental principles of how it works, its dependencies and requirements to ensure that it works correctly.

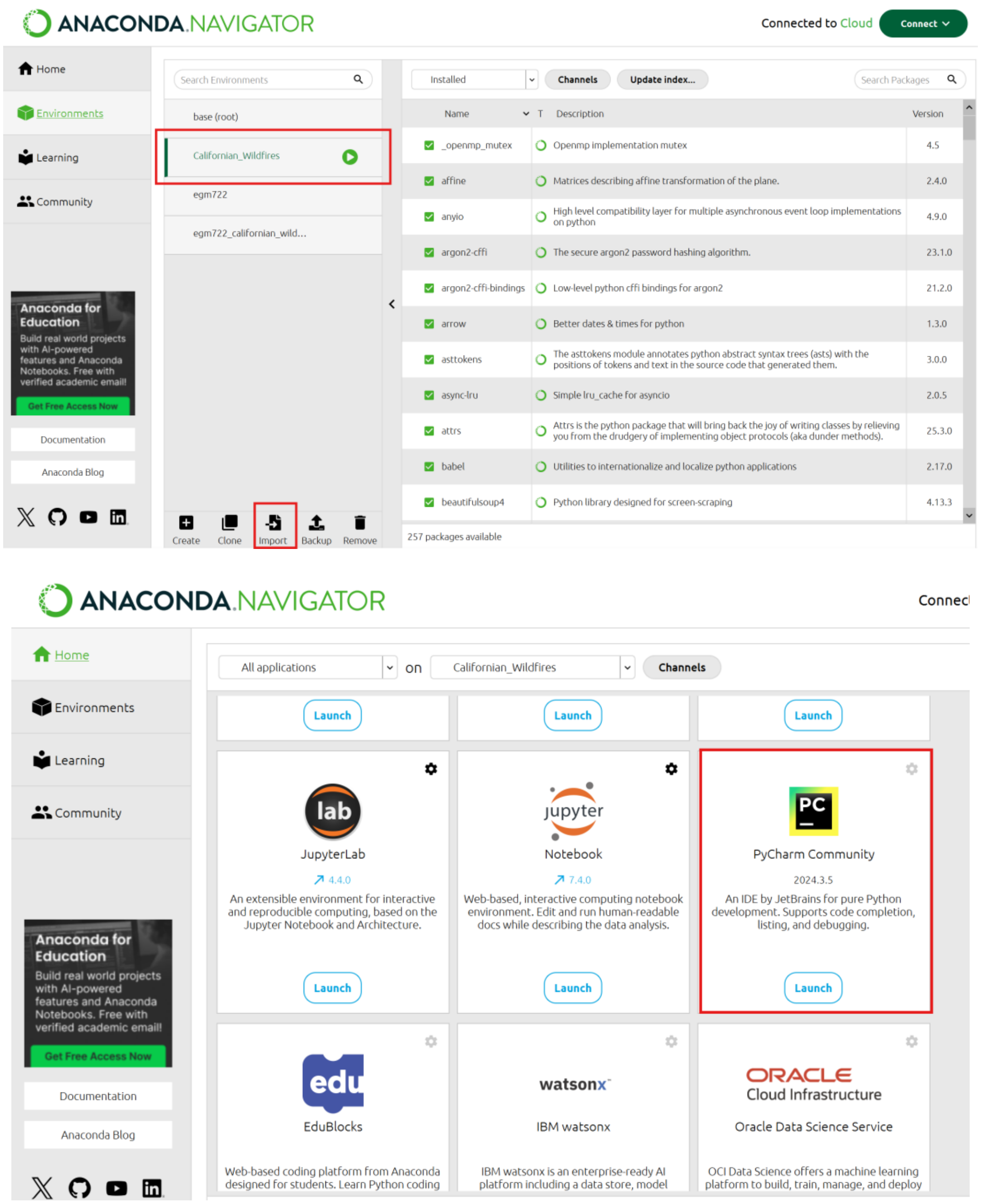
5.1 Fork / Clone: The required environment file, vector, python script and other files required for this program can be found within the public GitHub repository of Californian Wildfires Repository. Fork this repository into a local GitHub repository and name the repository appropriately such as "Wildfires in California".

5.2 Download Imagery: From USGS Earth Explorer download and save locally the following if not already competed previously.

- **Tile name**: LC09_L2SP_044034_20250409_20250411_02_T1
- **Source**: USGS Earth Explorer
- **Type**: OLI/TIRS C2 Level 2

5.3 Import Environment: The required environment file is found within GitHub and has the required dependencies, if at any point during the process of running the program such as module (package) issues, please refer to the troubleshooting section at the end of this document. The environment has been created using Notepad++ which could be used for future projects if required. This file is called Californian_Wildfires.yml. There is a prefix at the bottom which requires amending to the new location on the local system.

Import this environment into Anaconda Navigator. Depending on the system this can take a few minutes. *Figure 2: Import Environment* shows where this import should be done.



Name	T	Description	Version
_openmp_mutex		Openmp implementation mutex	4.5
affine		Matrices describing affine transformation of the plane.	2.4.0
anyio		High level compatibility layer for multiple asynchronous event loop implementations on python	4.9.0
argon2-cffi		The secure argon2 password hashing algorithm.	23.1.0
argon2-cffi-bindings		Low-level python cffi bindings for argon2	21.2.0
arrow		Better dates & times for python	1.3.0
asttokens		The asttokens module annotates python abstract syntax trees (asts) with the positions of tokens and text in the source code that generated them.	3.0.0
async-lru		Simple lru_cache for asyncio	2.0.5
attrs		Attrs is the python package that will bring back the joy of writing classes by relieving you from the drudgery of implementing object protocols (aka dunder methods).	25.3.0
babel		Utilities to internationalize and localize python applications	2.17.0
beautifulsoup4		Python library designed for screen-scraping	4.13.3

257 packages available

JupyterLab
4.4.0
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

Notebook
7.4.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

PyCharm Community
2024.3.5
An IDE by JetBrains for pure Python development. Supports code completion, listing, and debugging.
Launch

EduBlocks
Web-based coding platform from Anaconda designed for students. Learn Python coding

IBM watsonx
IBM watsonx is an enterprise-ready AI platform including a data store, model

ORACLE Cloud Infrastructure
Oracle Data Science Service
OCI Data Science offers a machine learning platform to build, train, manage, and deploy

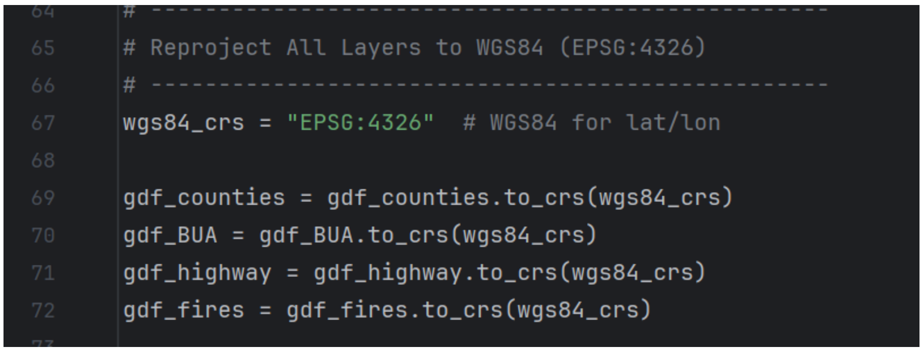
Anaconda for Education
Build real world projects with AI-powered features and Anaconda Notebooks. Free with verified academic email!
Get Free Access Now
Documentation
Anaconda Blog

Figure 2: Import Environment – Source: Anaconda Navigator interface showing where to import the environment file and how to launch PyCharm Community.

Following the completion of the upload ensure that PyCharm Community is installed and linked to this environment, this can be seen in *Fig 2*.

5.4 Data Management: Using the imagery tile downloaded as noted in the required datasets section, bring it into ArcPro. We know that the bands for Landsat 8-9 Operational Land Imager (OLI) and Thermal Infrared Sensor (TIRS) have its red band as 4, green as 3 and blue as 2. If we were to look at Near Infrared (NIR) we would be using band 5. But for this instance, the bands are defined as 4,3,2 (R, G, B). Using the toolbox, create a composite bands image. Ensure that you specify in the name when exporting that it’s an .tif image.

The Landsat has a projection of WGS84 UTM Zone 10N, this should match the vector that is downloaded or cloned from the GitHub repository. For the outputs, what will be seen later is that the coordinate system has changed to implement a Lat and Long grid. This is done within the script from line65 and can be seen in *Figure 3: Reproject Layers* where the script reprojects the layers to give a grid that is more sensible for the size of the area.



```
65    # Reproject All Layers to WGS84 (EPSG:4326)
66    # ---------------------------------------------------
67    wgs84_crs = "EPSG:4326"  # WGS84 for lat/lon
68
69    gdf_counties = gdf_counties.to_crs(wgs84_crs)
70    gdf_BUA = gdf_BUA.to_crs(wgs84_crs)
71    gdf_highway = gdf_highway.to_crs(wgs84_crs)
72    gdf_fires = gdf_fires.to_crs(wgs84_crs)
```

Figure 3: Reproject Layers – Source: Main_Program.py script found in GitHub repository showing reprojection of layer.

This also allows the user the ability to configure the extent through the use of Lat and Long rather than using the local coordinate system which requires the inputs in metres.

5.5 PyCharm Link: Launch PyCharm Community from Anaconda Navigator Once open import the repository from GitHub previously created, by selecting the menu Version Control System (VCS) and select get project from VCS and use the URL of the created GitHub repository previously created. The menu will now change as the VCS option would be replaced with a Git menu that provides the tools to commit, push, pull etc. At this stage there is now a link between PyCharm, and the repository previously created by forking over the files in GitHub. *Figure 4: PyCharm Link* shows what this menu should look like.

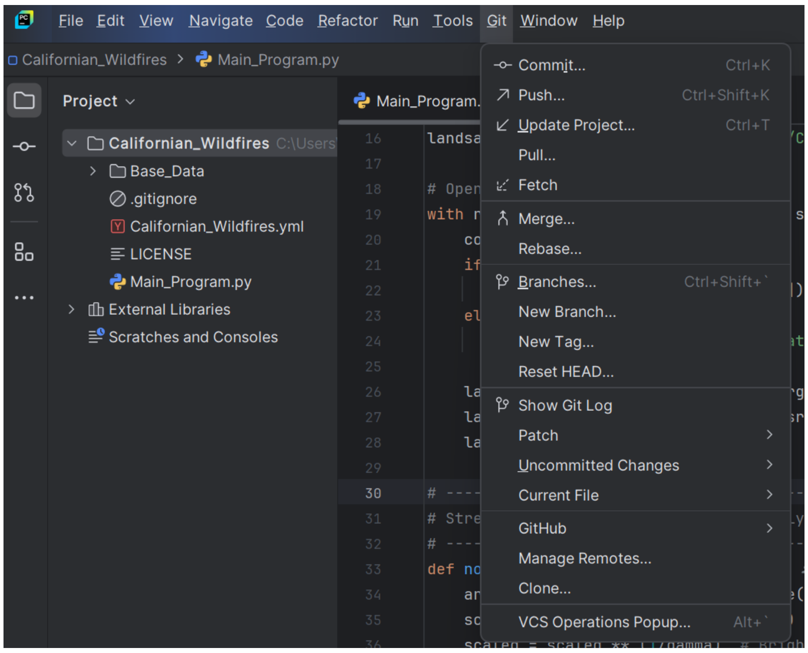
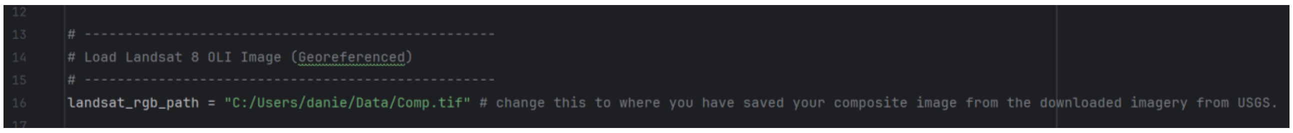


Figure 4: PyCharm Link – Source: PyCharm interface showing menu options once link to GitHub repository has been made.

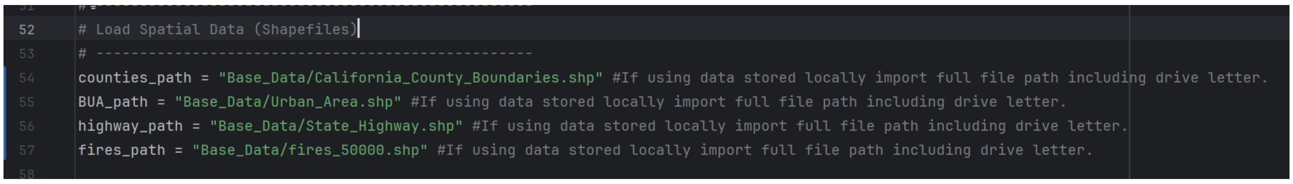
5.6 Modify Script: Within PyCharm there will be a requirement to adjust the script to point to where the Landsat imagery has been downloaded on the local system. This can be found at line 16 where the file path name will need to be amended. *Figure 5: Modify Imagery File Path* shows what this will look like.



```
13  # -------------------------------------------------
14  # Load Landsat 8 OLI Image (Georeferenced)
15  # -------------------------------------------------
16  landsat_rgb_path = "C:/Users/danie/Data/Comp.tif" # change this to where you have saved your composite image from the downloaded imagery from USGS.
```

Figure 5: Modify Imagery File Path – Source: Program.py script found in GitHub repository displaying where user is to input source of imagery file highlighted with green text.

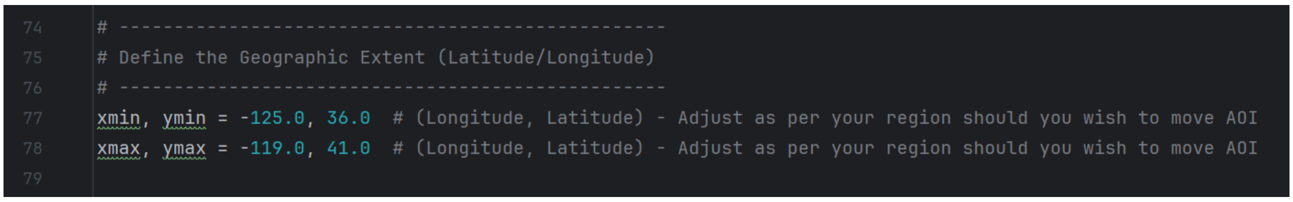
There may be a requirement to amend the file paths of the vector if more current data has been downloaded. To amend this simply navigate to line 54 and import the latest version of the data. *Figure 6: Shapefile Modification* shows what this will look like. Please note that where the path starts with “Base_Data” this is referring to the Base_Data folder found within the GitHub repository. If data is downloaded to the local system a full file path is required to the data location e.g. C:/Users/Data/<fires_path>.



```
52  # Load Spatial Data (Shapefiles)
53  # -------------------------------------------------
54  counties_path = "Base_Data/California_County_Boundaries.shp" #If using data stored locally import full file path including drive letter.
55  BUA_path = "Base_Data/Urban_Area.shp" #If using data stored locally import full file path including drive letter.
56  highway_path = "Base_Data/State_Highway.shp" #If using data stored locally import full file path including drive letter.
57  fires_path = "Base_Data/fires_50000.shp" #If using data stored locally import full file path including drive letter.
```

Figure 6: Shapefile Modification – Source: Program.py script found in GitHub repository displaying shapefile paths highlighted in green.

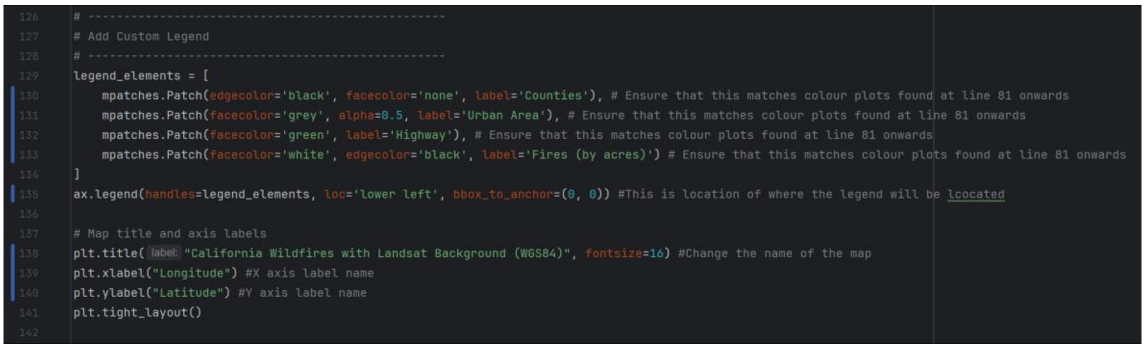
5.7 Modify Map: There may be a requirement to modify how the output looks or what the extent of the output covers. This can be modified at line 75 onwards as seen in *Figure 7: Modify Extent* which shows that through simply inputting the min and max x, y coordinates (Lat and Long) the map extent will change the outputs.



```
74  # -------------------------------------------------
75  # Define the Geographic Extent (Latitude/Longitude)
76  # -------------------------------------------------
77  xmin, ymin = -125.0, 36.0  # (Longitude, Latitude) - Adjust as per your region should you wish to move AOI
78  xmax, ymax = -119.0, 41.0  # (Longitude, Latitude) - Adjust as per your region should you wish to move AOI
```

Figure 7: Modify Extent - Source: Program.py script found in GitHub repository displaying where extents can be modified, highlighted in blue text.

The marginalia of the outputs can be modified at the latter stage of the script such as the legend elements which can be found from line 127, the title of the outputs found from line 137, these can be seen in *Figure 8: Modify Marginalia*.



```
# --------------------------------------------------
# Add Custom Legend
# --------------------------------------------------
legend_elements = [
    mpatches.Patch(edgecolor='black', facecolor='none', label='Counties'), # Ensure that this matches colour plots found at line 81 onwards
    mpatches.Patch(facecolor='grey', alpha=0.5, label='Urban Area'), # Ensure that this matches colour plots found at line 81 onwards
    mpatches.Patch(facecolor='green', label='Highway'), # Ensure that this matches colour plots found at line 81 onwards
    mpatches.Patch(facecolor='white', edgecolor='black', label='Fires (by acres)') # Ensure that this matches colour plots found at line 81 onwards
]
ax.legend(handles=legend_elements, loc='lower left', bbox_to_anchor=(0, 0)) #This is location of where the legend will be lcocated

# Map title and axis labels
plt.title( label: "California Wildfires with Landsat Background (WGS84)", fontsize=16) #Change the name of the map
plt.xlabel("Longitude") #X axis label name
plt.ylabel("Latitude") #Y axis label name
plt.tight_layout()
```

Figure 8: Modify Marginalia- Source: Program.py script found in GitHub repository showing where the legend and other marginalia can be modified.

5.8. Results

Once executed, the script produces the following outputs and can be located on the local system where the GitHub folder sits e.g. C:/Users/<name>/GitHub/<Repoistory Name>/:

- A spatially accurate .tif file visualising wildfire proximity.
- A presentation ready .png output map.
- Layers illustrating urban areas, wildfire perimeters, and evacuation routes.

These products allow users to assess fire threats in relation to the human population and infrastructure and can be adapted for other regions or updated datasets. *Figure 9: PNG Output* shows the expected output of the PNG file and *Figure 10: Tif for PRO with marginalia* shows the expected output of the Tif that is ready for import into GIS software such as ArcGIS Pro.

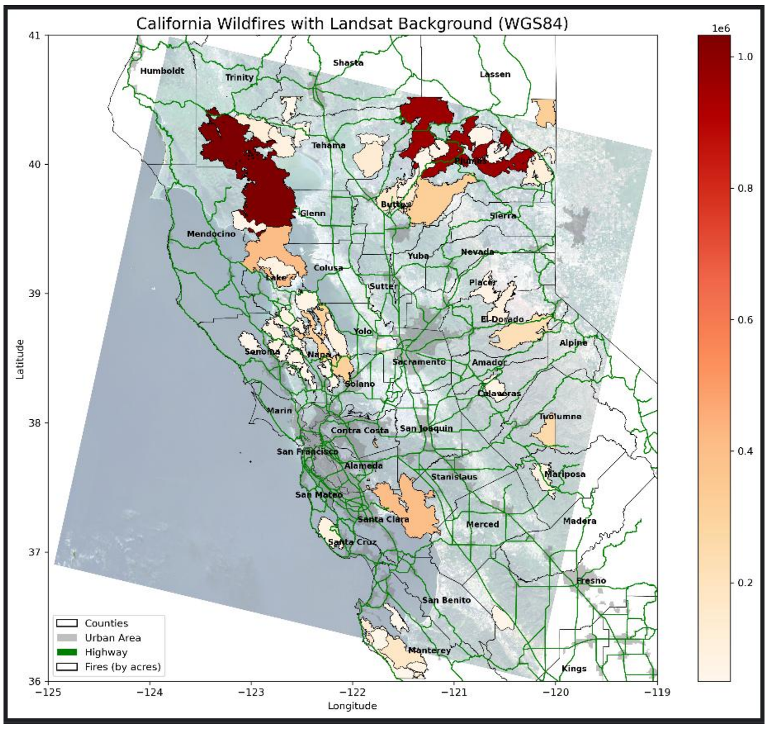


Figure 9: PNG Output -Source: Taken from the result of running the script and producing a .Png file.

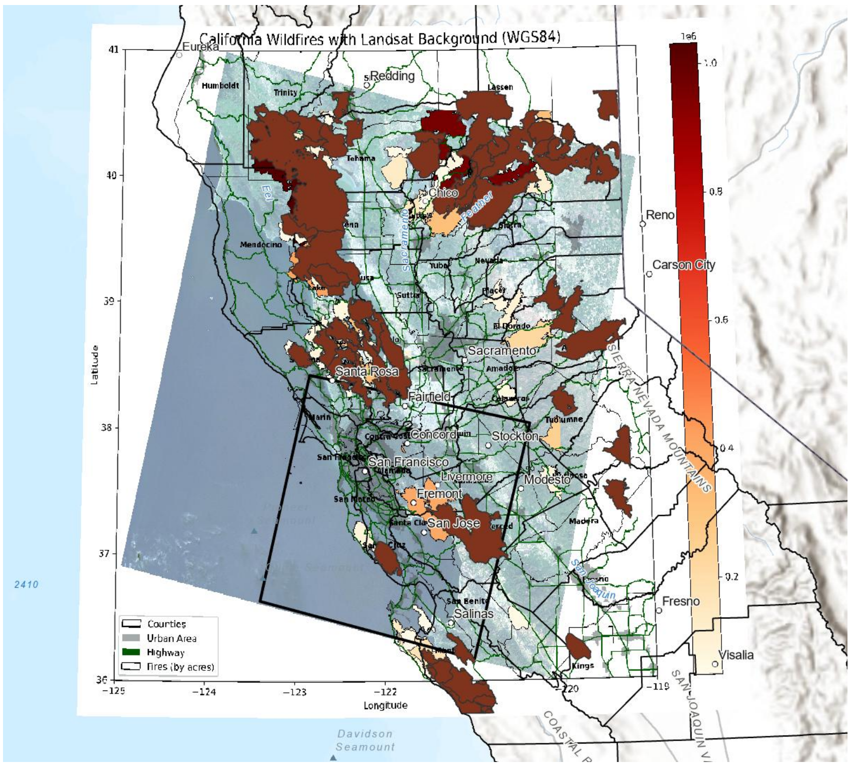


Figure 10: Tif for PRO with marginalia -Source: Taken from the result of running the script and producing a .Tif file compatible with other GIS software.

5.8.1 Branches: According to Walrad and Strom (2002), the odds of updating something within a script without issues are poor and therefore branching is essential for version management. The current script runs and creates marginalia for both files. Considering that the Tif's purpose is to be imported into other GIS software it may be that the marginalia is not required or wanted. To remove this a branch has been created in the GitHub Repository that achieves this. Simply commit that change to the main branch from the branch for the desired output without marginalia.

When committed to the main branch the marginalia will be removed from the outputs. *Figure 11: Removal of marginalia* shows what the expected output of this commit will look like in ArcPro.

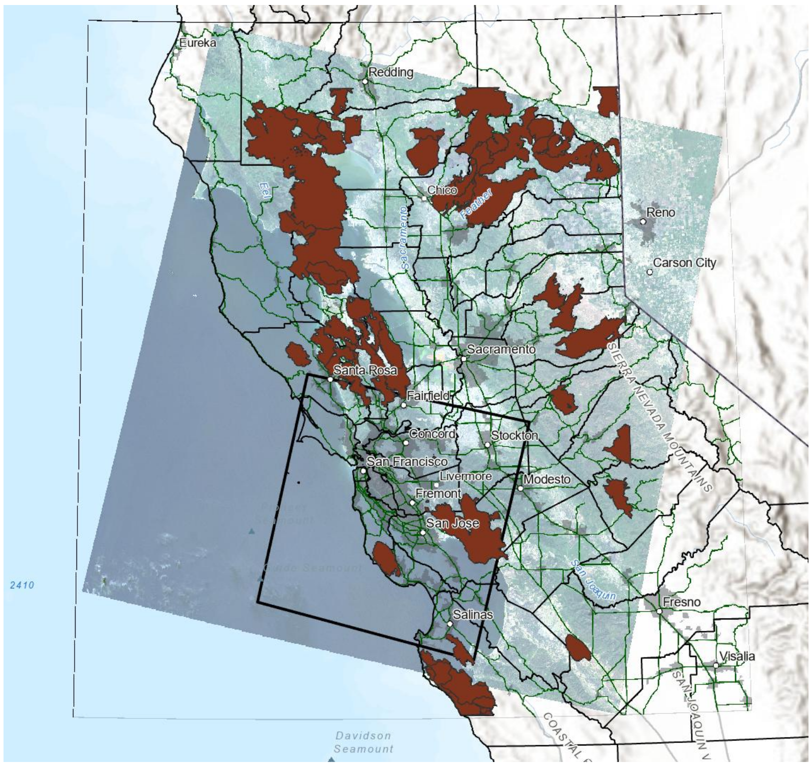


Figure 11: Removal of marginalia-Source: Taken from the result of running the script and producing a .Tif file without surrounding information, compatible with other GIS software.

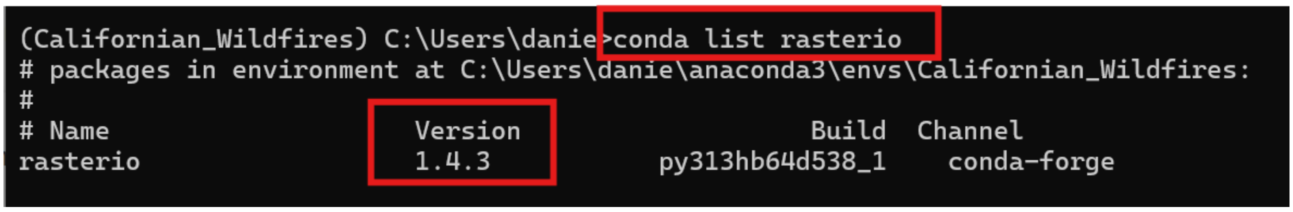
There is a known issue which is alluded to in the README file on GitHub that states the symbology of the fires layer changes and that it hasn't been rectified.

6. Troubleshooting

The most common error that is likely to occur will be that a module is not found such as rasterio. Before going through the installation again, ensure that it is actually present in the environment. Through Anaconda Navigator open up a CMD window, ensure that the correct environment is activated and run the following:

- Conda list <package>

The package is to be replaced by the package that is believed to be missing. *Figure 12: Package Search* shows this with the rasterio package.



```
(Californian_Wildfires) C:\Users\danie>conda list rasterio
# packages in environment at C:\Users\danie\anaconda3\envs\Californian_Wildfires:
#
# Name                    Version                   Build  Channel
rasterio                  1.4.3           py313hb64d538_1    conda-forge
```

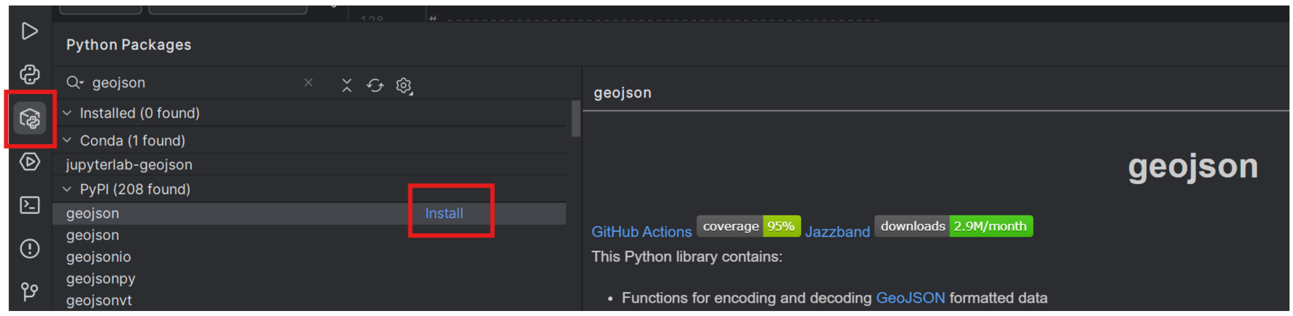
Figure 12; Package Search-Source: Taken from Anaconda CMD showing method to examine installed packages.

What this also provides it the version number which is worth noting if the script is trying to access an older version. If the package isn't present, within the same terminal input the following:

- conda install <Package>

A version number can also be implemented here after the package name by inputting a version number directly after the package name without spaces e.g. conda install rasterio=1.4.3

Alternatively, within PyCharm select the Python Packages panel located at the bottom left of the window and search for the required package e.g. geojson and install. *Figure 13: Package Install* shows where this located and where the instal button is found.



Python Packages

geojson

Installed (0 found)
Conda (1 found)
jupyterlab-geojson
PyPI (208 found)
geojson — Install
geojson
geojsonio
geojsonpy
geojsonvt

geojson

geojson

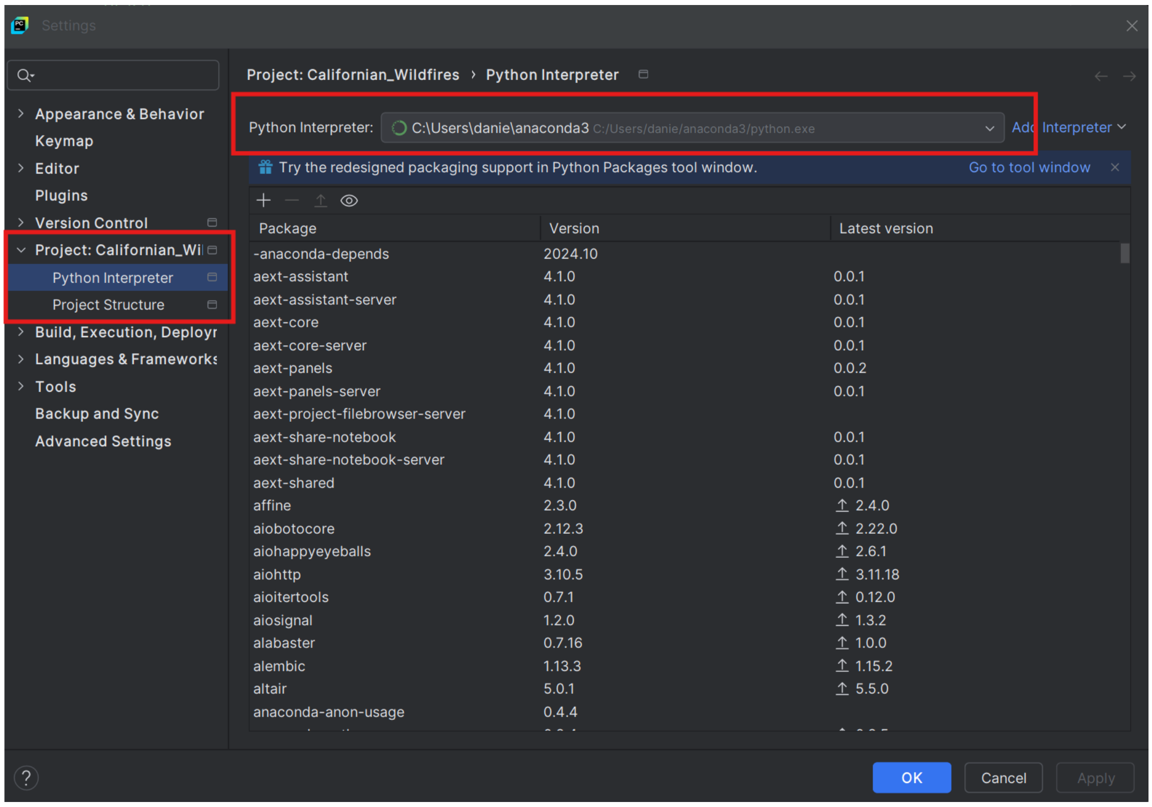
GitHub Actions | coverage 95% | Jazzband | downloads 2.9M/month

This Python library contains:

- Functions for encoding and decoding GeoJSON formatted data

Figure 13: Package Install - Source: Taken from PyCharm interface showing where packages can be installed within the IDE.

Another error could be that the interpreter isn't pointing to the correct location i.e. the anaconda3 python interpreter. To fix this open up the setting (Ctrl Alt +S) panel and ensure that the interpreter for the program is pointing to the right location. This can be here in *Figure 14: Python Interpreter.*



Project: Californian_Wildfires › Python Interpreter

Python Interpreter: C:\Users\danie\anaconda3 C:/Users/danie/anaconda3/python.exe — Add Interpreter

Try the redesigned packaging support in Python Packages tool window. Go to tool window

Package	Version	Latest version
-anaconda-depends	2024.10	
aext-assistant	4.1.0	0.0.1
aext-assistant-server	4.1.0	0.0.1
aext-core	4.1.0	0.0.1
aext-core-server	4.1.0	0.0.1
aext-panels	4.1.0	0.0.2
aext-panels-server	4.1.0	0.0.1
aext-project-filebrowser-server	4.1.0	
aext-share-notebook	4.1.0	0.0.1
aext-share-notebook-server	4.1.0	0.0.1
aext-shared	4.1.0	0.0.1
affine	2.3.0	↑ 2.4.0
aiobotocore	2.12.3	↑ 2.22.0
aiohappyeyeballs	2.4.0	↑ 2.6.1
aiohttp	3.10.5	↑ 3.11.18
aioitertools	0.7.1	↑ 0.12.0
aiosignal	1.2.0	↑ 1.3.2
alabaster	0.7.16	↑ 1.0.0
alembic	1.13.3	↑ 1.15.2
altair	5.0.1	↑ 5.5.0
anaconda-anon-usage	0.4.4	

Figure 14: Python Interpreter - Source: Taken from PyCharm interface showing where to ensure the correct interpreter is being used within the IDE.

7. References

Hu, Y., Zhang, J., Bai, X., Yu, S. and Yang, Z., 2016. Influence analysis of GitHub repositories. *SpringerPlus*, 5, Article 1268. https://doi.org/10.1186/s40064-016-2897-7

Palinkas, L.A., 2020. The California Wildfires. In: L.A. Palinkas, ed. *Global Climate Change, Population Displacement, and Public Health: The Next Wave of Migration*. 1st ed. Cham: Springer Nature Switzerland AG, pp. 53–67.

Walrad, C. and Strom, D., 2002. The importance of branching models in SCM.*Computer*, *35*(9), pp.31-38.

Words – 2,064