Working with big data with ArcGIS API for Python

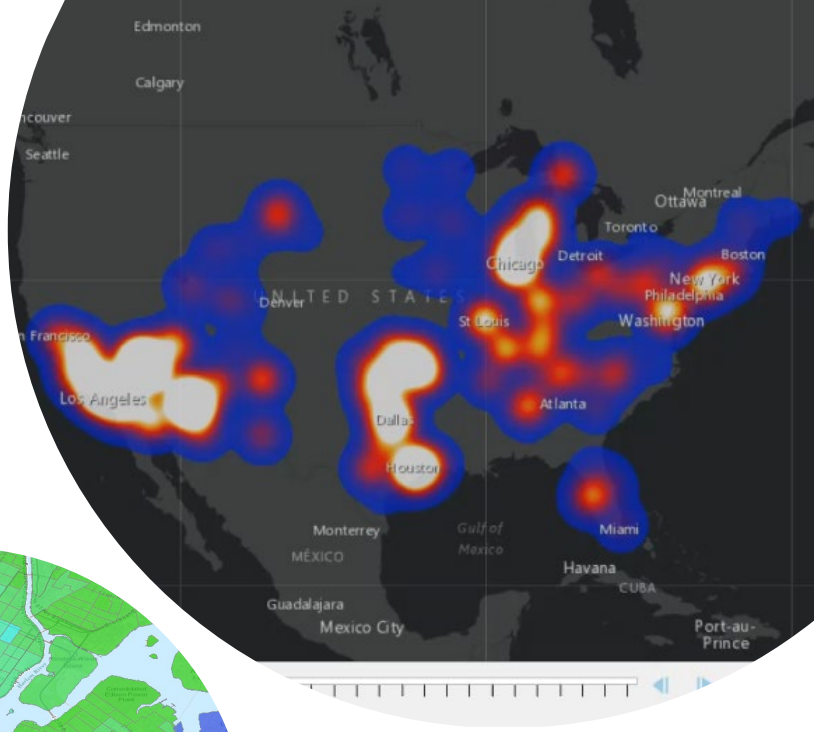
Sarah Ambrose and Priyanka Tuteja

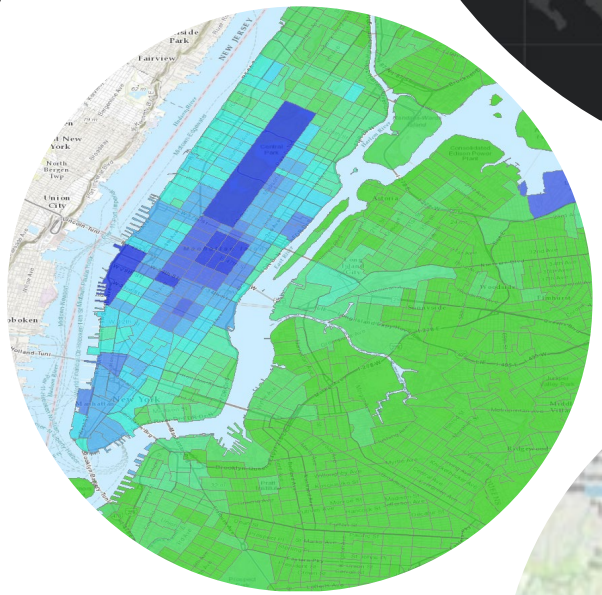
2021 ESRI
DEVELOPER SUMMIT

What is GeoAnalytics Server?

A toolbox that *parallelizes* computation to quickly analyze large amounts of vector and tabular data

A collection of analysis tools to identify *patterns*, *relationships*, *anomalies* and *incidents* in large amounts of data across *space and time*





How is it faster?

What kind of analysis can I run?

- **Which stationary pressure sensors in my pipe network have experienced anomalous events in the past 24 hours? Where are there hot spots of anomalous events?**
- Where have my delivery trucks traveled and where is the highest density of unique delivery truck paths? Where do delivery trucks travel the slowest?
- Where and when are events happening close together in space and time?

What kind of analysis can I run?

- Which stationary pressure sensors in my pipe network have experienced anomalous events in the past 24 hours? Where are there hot spots of anomalous events?
- **Where have my delivery trucks traveled and where is the highest density of unique delivery truck paths? Where do delivery trucks travel the slowest?**
- Where and when are events happening close together in space and time?

What kind of analysis can I run?

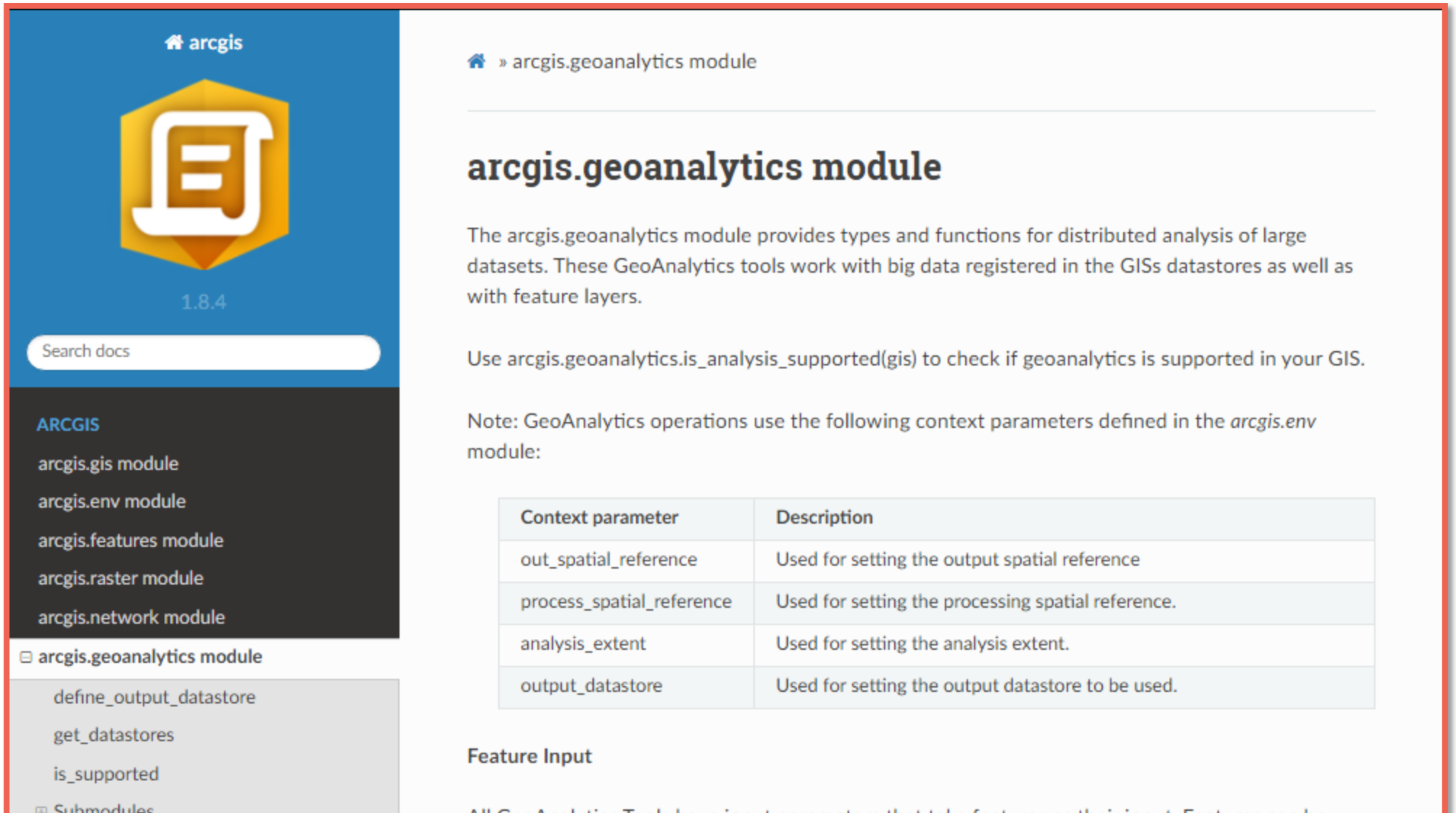
- Which stationary pressure sensors in my pipe network have experienced anomalous events in the past 24 hours? Where are there hot spots of anomalous events?
- Where have my delivery trucks traveled and where is the highest density of unique delivery truck paths? Where do delivery trucks travel the slowest?
- **Where and when are events happening close together in space and time?**

ArcGIS API for Python

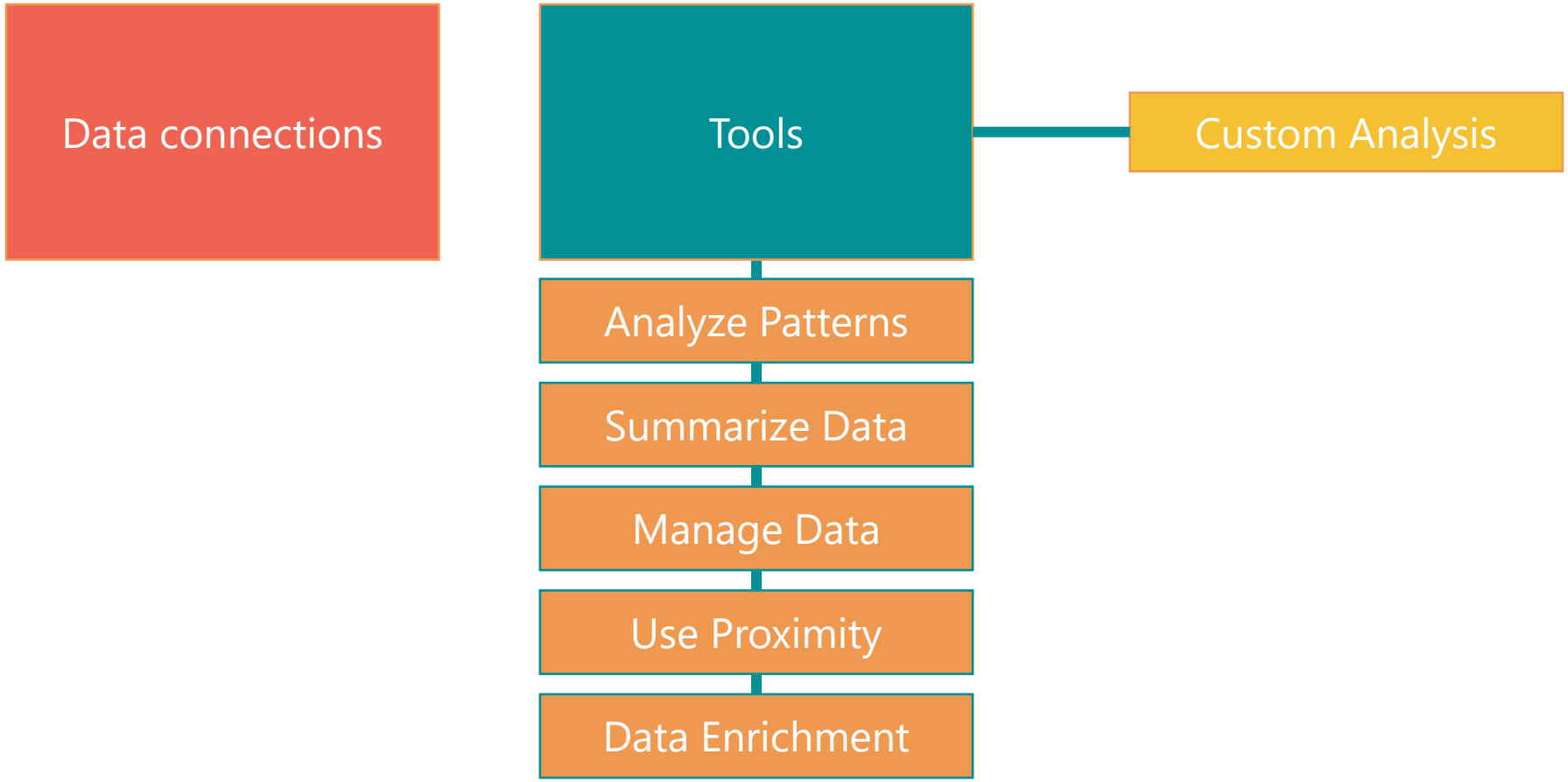
- Python library for spatial analysis, mapping, and GIS
- Powered by WebGIS

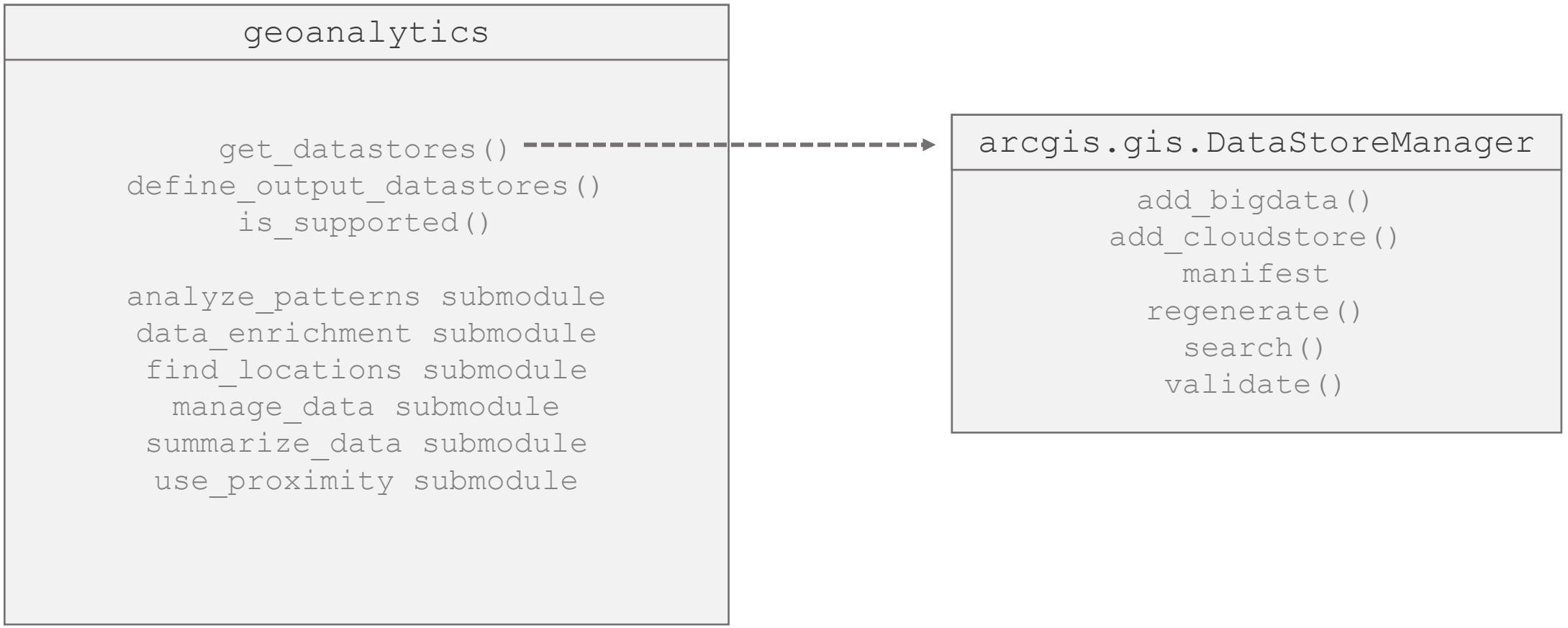
Made for automation and data science

How do they come together?



What do you get?




geoanalytics
get_datastores()
define_output_datastores()
is_supported()
analyze_patterns submodule
data_enrichment submodule
find_locations submodule
manage_data submodule
summarize_data submodule
use_proximity submodule
arcgis.gis.DataStoreManager
add_bigdata()
add_cloudstore()
manifest
regenerate()
search()
validate()

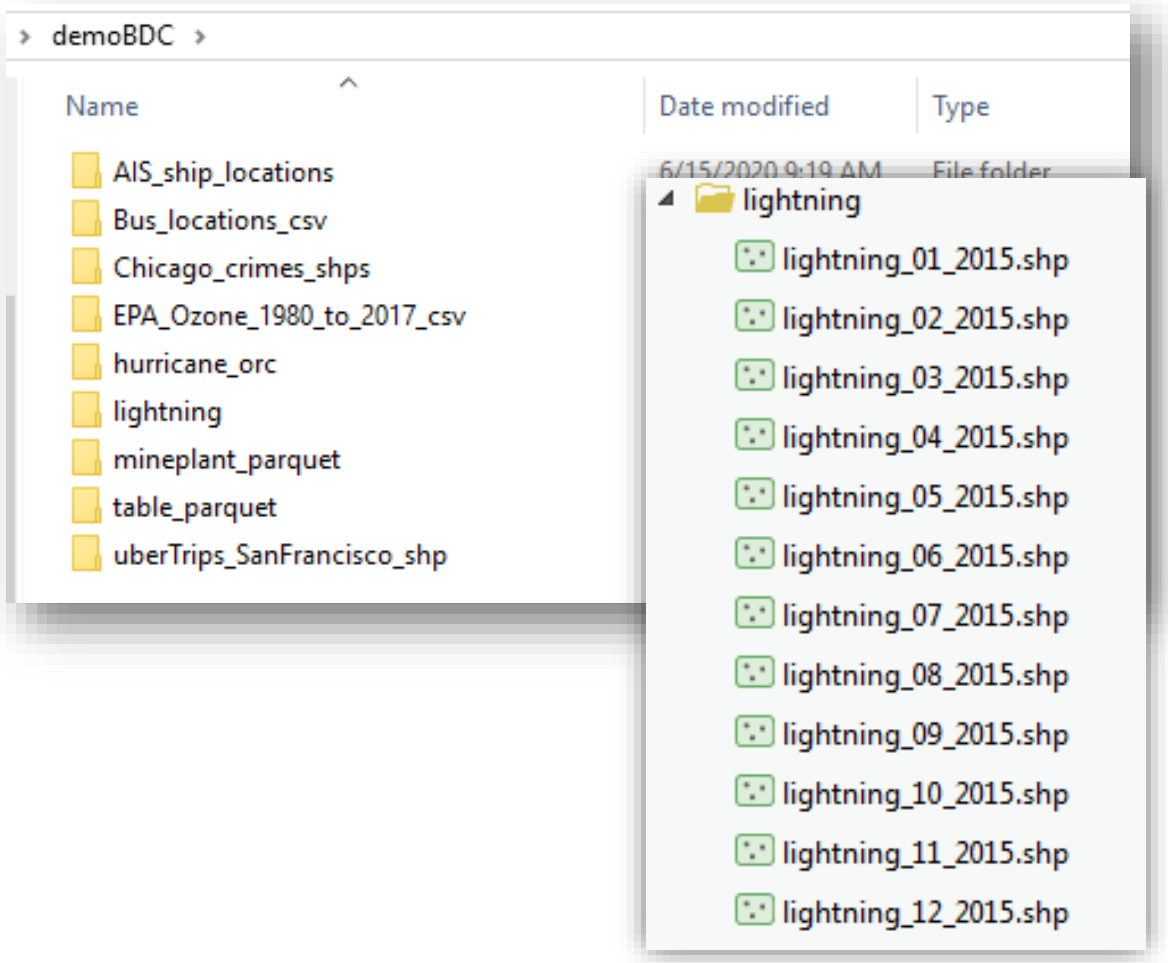
Big data file shares

Read directly from files stores in:

- Hive
- HDFS
- Shared folders
- Cloud Stores

Supported file types include:

- Delimited files
- Shapefiles
- Parquet
- ORC



Demo

Priyanka Tuteja

summarize_data

aggregate_points()
build_multivariable_grid()
describe_dataset()
join_features()
reconstruct_tracks()
summarize_attributes()
summarize_within()
summarize_center_and_dispersion()

use_proximity

create_buffers()
trace_proximity_events()

find_locations

detect_incidents()
find_dwell_locations()
find_similar_locations()
geocode_locations()

manage_data

append_data()
calculate_field()
clip_layer()
copy_to_data_store()
dissolve_boundaries()
merge_layers()
overlay_data()
summarize_center_and_dispersion()
run_python_script()

analyze_patterns

calculate_density()
create_space_time_cube()
find_hot_spots()
find_point_clusters()
forest()
glr()
gwr()

data_enrichment

enrich_from_grid()
calculate_motion_statistics()

Access and use PySpark with GeoAnalytics Server

Use Run Python Script to execute distributed analysis

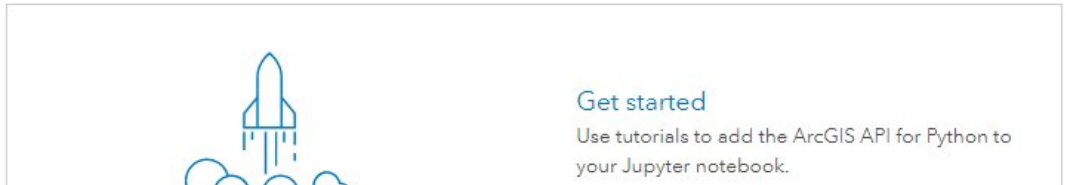
- Run a custom python script on your GeoAnalytics Server site
- Use other python functionality and distribute analysis across your site
- Create an analysis pipeline to chain GeoAnalytics tools together
- Use pyspark (ml, sql) and data frames

Demo

Priyanka Tuteja

Tips and Tricks

Documentation: 3 ways to learn

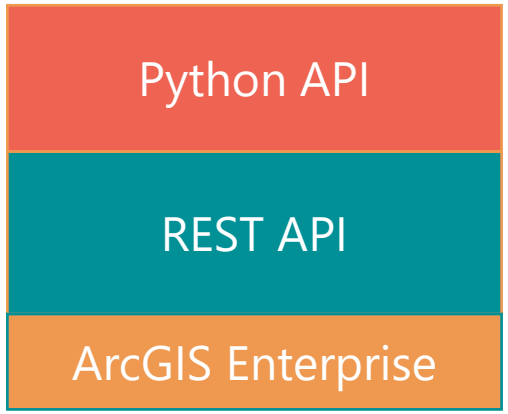


Tips and Tricks

Most functions are built on REST endpoints

- You can use the REST API documentation if something isn't clear in the ArcGIS API for Python doc
- You can test things out in REST or the UI
- More concept help in Enterprise documentation

Ask questions on GeoNet



Tips and Tricks

You can use the future option:

future	Optional boolean. If True, a GPJob is returned instead of results. The GPJob can be queried on the status of the execution.

class **arcgis.geoprocessing.GPJob**(*future, gptool, jobid, task_url, gis, notify=False*)

Represents a Single Geoprocessing Job. The *GPJob* class allows for the asynchronous operation of any geoprocessing task. To request a GPJob task, the code must be called with *future=True* or else the operation will occur synchronously. This class is not intended for users to call directly.

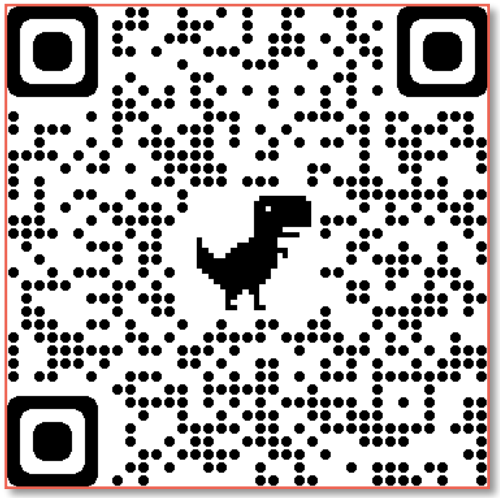
Argument	Description
future	Required ccurrent.futures.Future. The async object created by the geoprocessing (GP) task.
gptool	Required Layer. The Geoprocessing Service
jobid	Required String. The unique ID of the GP Job.
task_url	Required String. The URL to the GP Task.
gis	Required GIS. The GIS connection object
notify	Optional Boolean. When set to True, a message will inform the user that the geoprocessing task has completed. The default is False.

Why?

- Updates on the status of the job
- Returns a GP job object that can be run asynchronously
- You can cancel the job

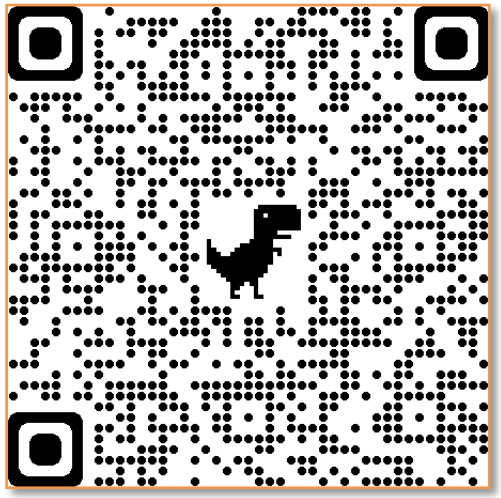
Thank you!

See the big data guide here:



See the REST API help here:

Ask questions on GeoNet



Please provide your feedback for this session by clicking on the session survey link directly below the video.