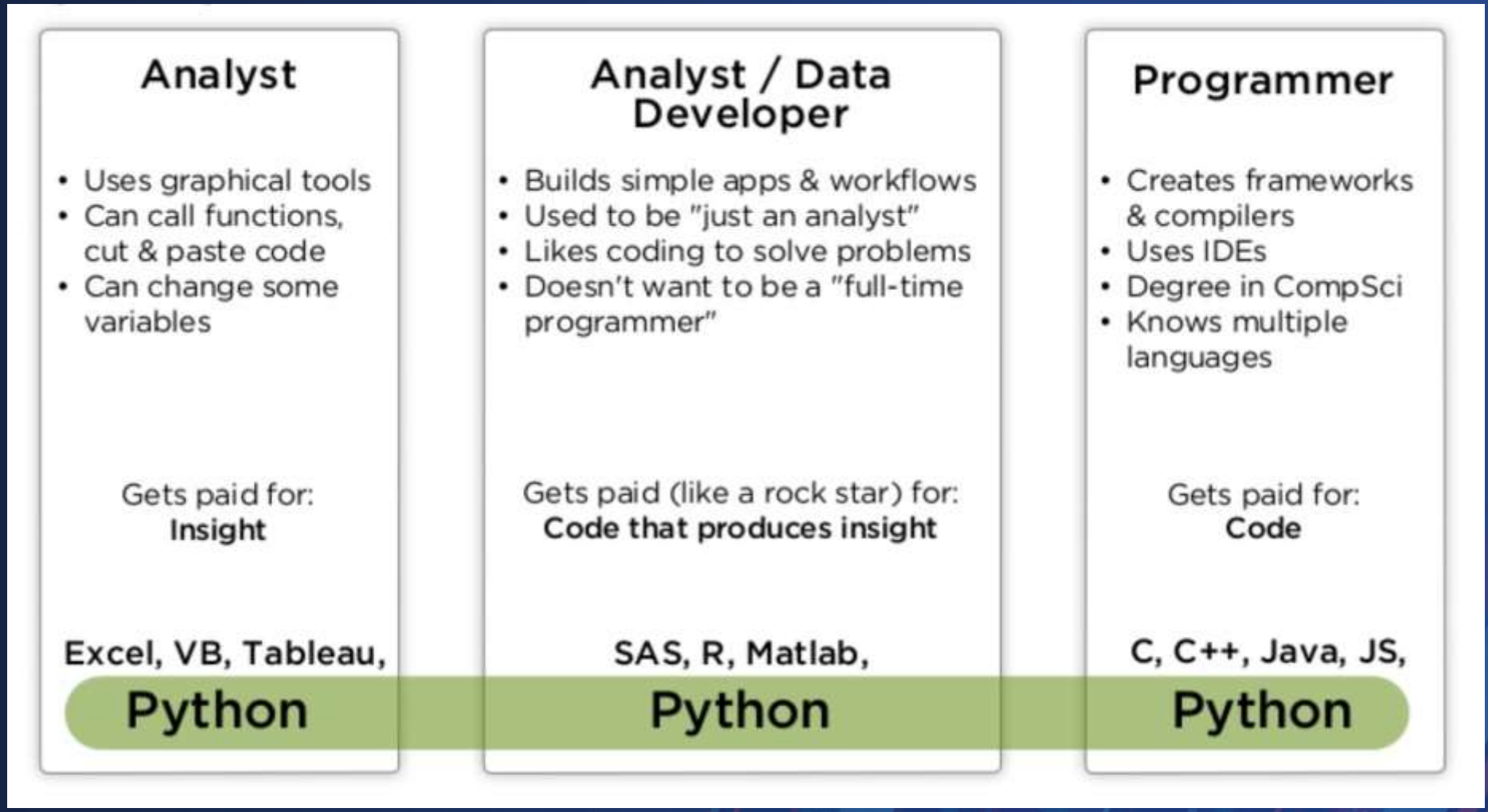
ArcGIS API for Python for ANALYSTS and DATA SCIENTISTS

ALBERTO NIETO
ANDREW CHAPKOWSKI - @achapkowski

2019 ESRI
FEDERAL GIS CONFERENCE
WASHINGTON, DC

Introduction

Analysts, Data Scientists and Developers



Source: Gateway Data Sciences Courses Reach Enrollment Milestones

Analysis

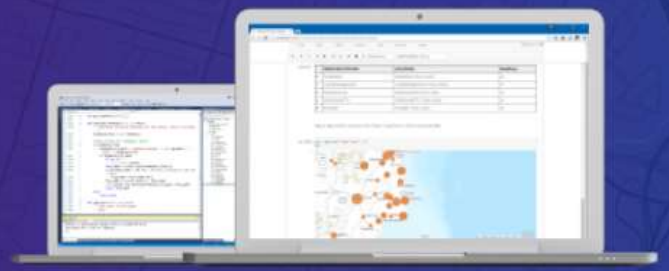
ArcGIS for Developers | Get Started | Documentation | Features | Pricing | Support | Sign In

Scripting and Automation / **ArcGIS API for Python (1.3)**

ArcGIS API for Python

Install the API

Version 1.3 · December 2017



Home | Guide | Sample Notebooks | API Reference | Community

A powerful Python library for spatial analysis, mapping and GIS

ArcGIS API for Python is a Python library for working with maps and geospatial data, powered by web GIS. It provides simple and efficient tools for sophisticated vector and raster analysis, geocoding, map making, routing and directions, as well as for organizing and managing a GIS with users, groups and information items. In addition to working with your own data, the library enables access to ready to use maps and curated geographic data from Esri and other authoritative sources. It also integrates well with the scientific Python ecosystem and includes rich support for Pandas and Jupyter notebook.

Install the API | Get started | View samples

Understand your GIS

This "hello world" style notebook shows how to get started with the GIS and visualize its contents.

› Get started with the GIS class

Manage your GIS

The ArcGIS API for Python provides APIs and samples for ArcGIS Online administrators to manage their online organization.

› Clone a portal

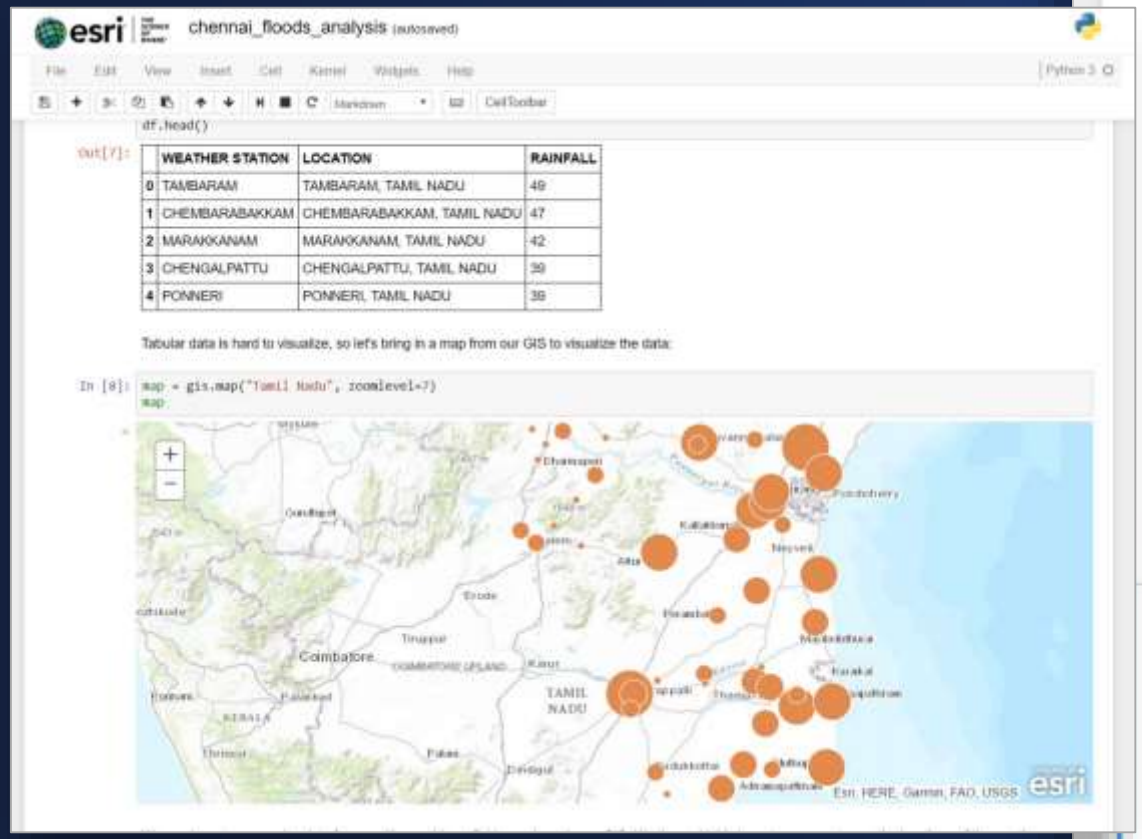
Perform Spatial Analysis

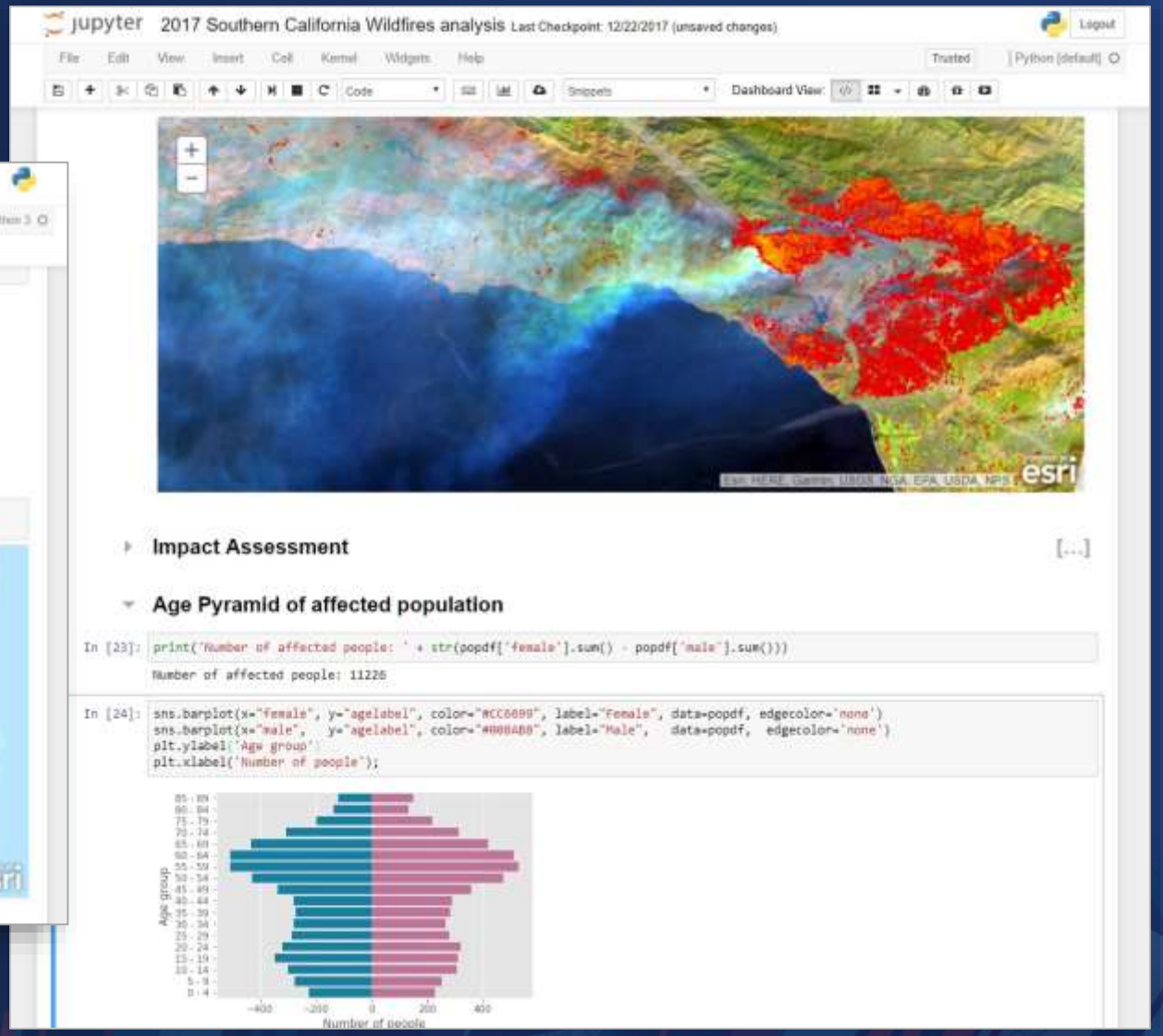
Call sophisticated spatial analysis tools that wo online content, using a few lines of code.

› Chennai floods analysis

geo enrichment
widgets
geometry
mapping
geocoding
geo processing
gis
geo analytics
realtime
schematics
Spatial DataFrame
raster
network
features
env

ArcGIS + Jupyter = ♥





It all starts with your GIS

In [1]:
```
from arcgis.gis import GIS
```

In [2]:
```
gis = GIS('https://deldev.maps.arcgis.com', 'demo_deldev')
```

In [3]:
```
enterprise = GIS('https://python.playground.esri.com/portal', 'arcgis_python',
```

Search for content

In [4]:
```
items = gis.content.search('San Diego')
```

In [5]:
```
for item in items:
    display(item)
```

Places to see in San Diego
Places to see in San Diego

Feature Collection by deldev
Last Modified: July 01, 2017
0 comments, 512 views

San Diego attractions web map
Esri Story Maps team member and San Diego resident Rupert Essinger selects some places you might enjoy.

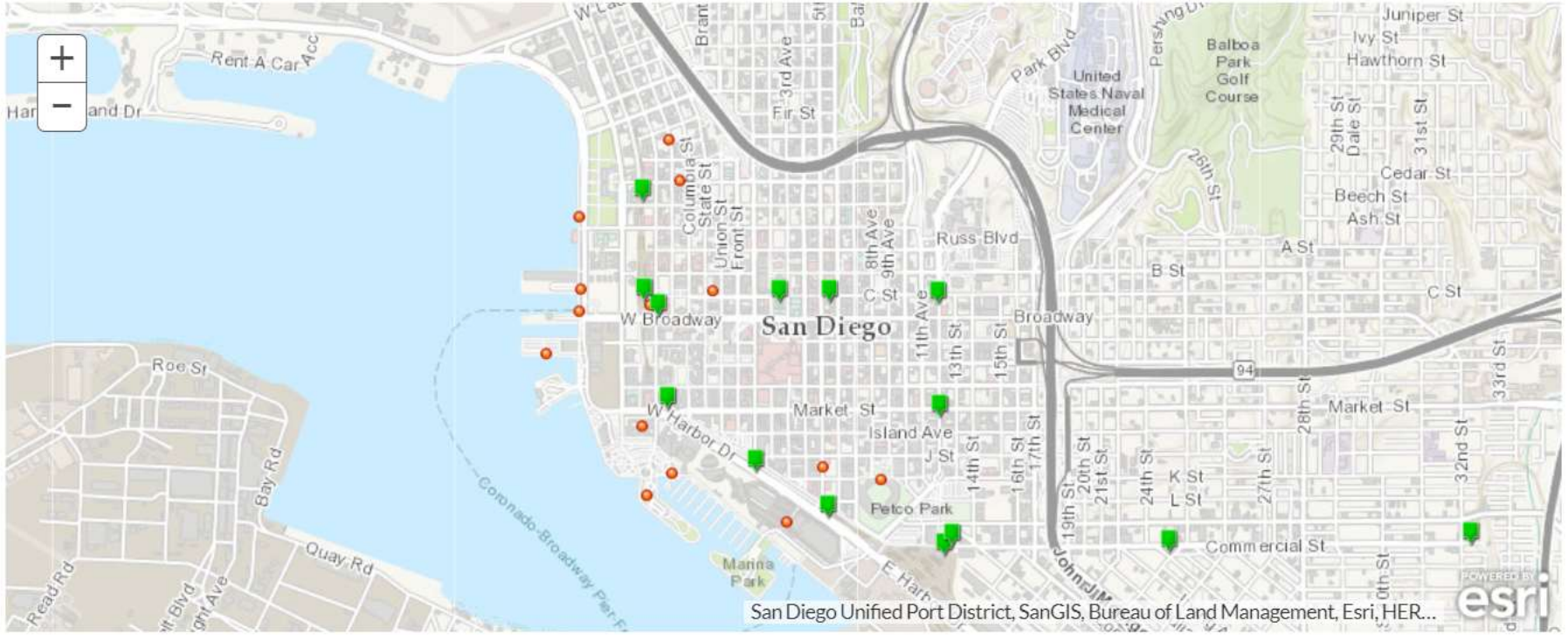
Web Map by deldev
Last Modified: July 01, 2017
0 comments, 3 views

San Diego Trolley stations
San Diego Trolley stations

Feature Collection by deldev
Last Modified: June 23, 2017

Visualize layers on map widget

In [7]:
```
sdmap = gis.map('San Diego', zoomlevel=14)
sdmap
```



In [8]:
```
sdmap.add_layer(sd_attractions)
```

In [9]:
```
sdmap.add_layer(trolley_stations)
```

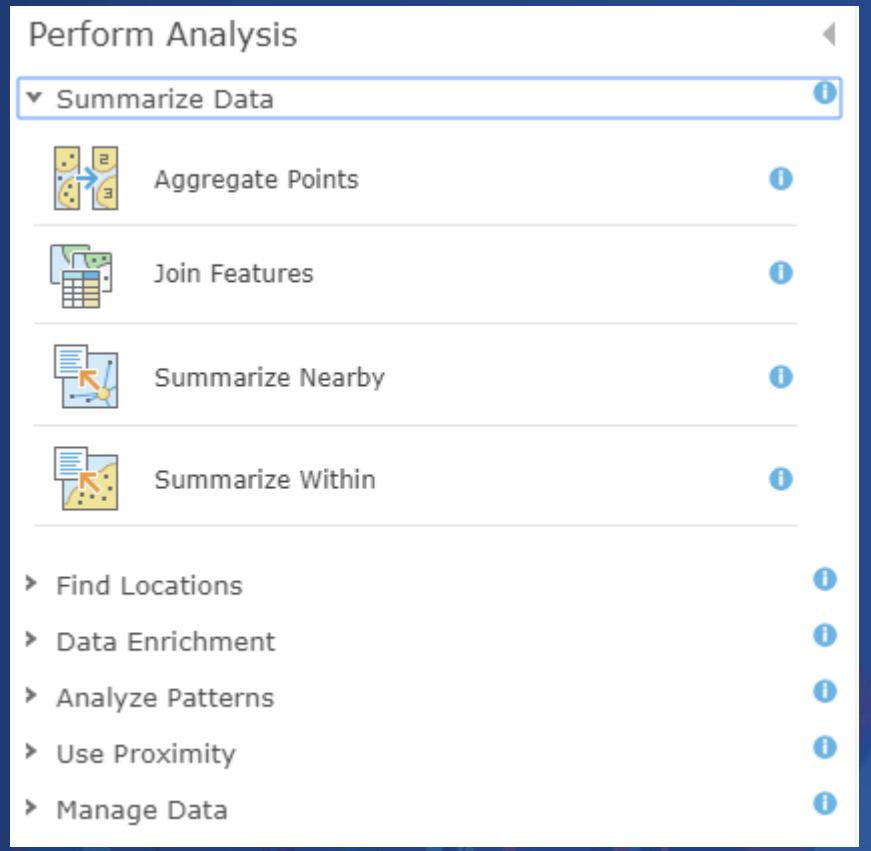
Spatial Analysis

Discover relationships, patterns and trends in data

`arcgis.feature` submodules

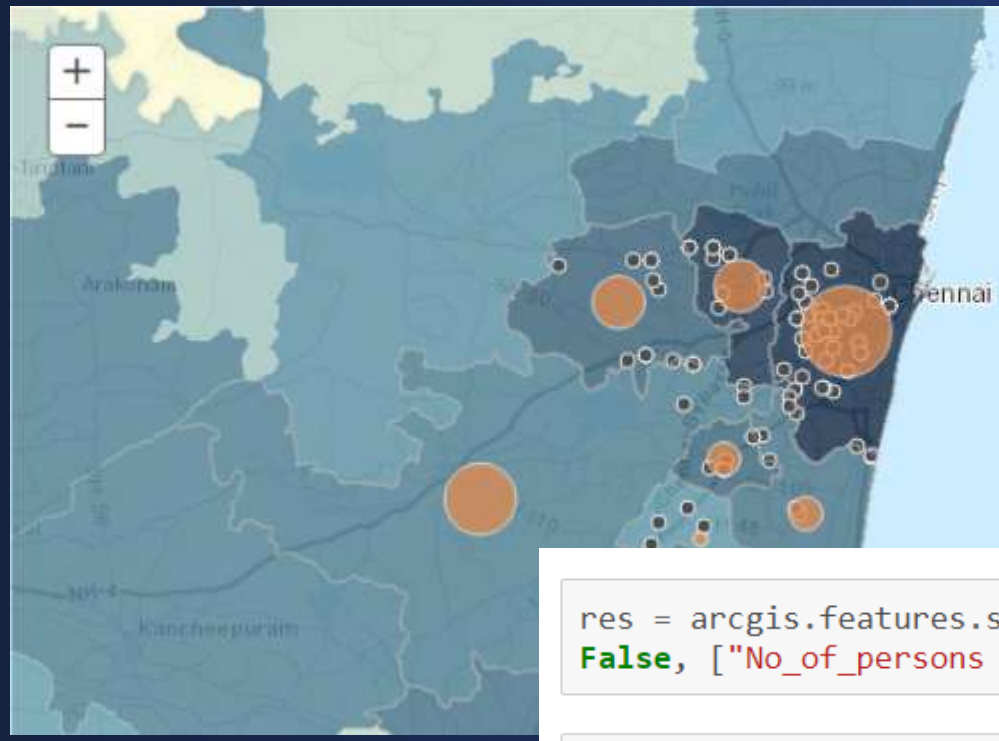
Spatial Analysis Tools

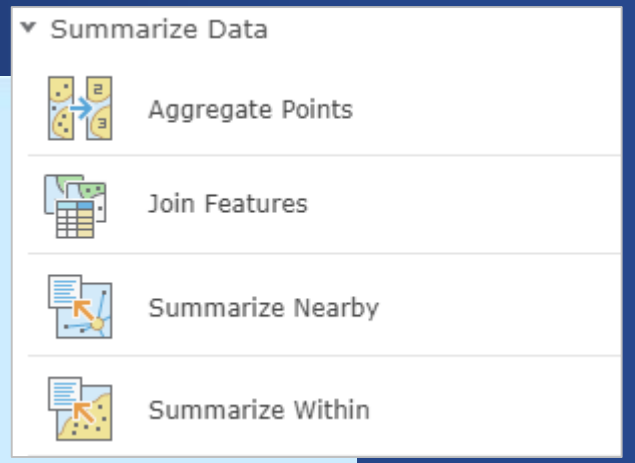
- Summarize Data
- Find Locations
- Data Enrichment
- Analyze Patterns
- Use Proximity
- Manage Data



Summarize Data

Calculate summary statistics for features and attributes






```
res = arcgis.features.summarize_data.aggregate_points(relief_centers, chennai_pop_featurelayer,
False, ["No_of_persons Sum"])
```

```
aggr_lyr = res['aggregated_layer']
```

```
reliefmap.add_layer(aggr_lyr, { "renderer": "ClassedSizeRenderer",
"field_name":"SUM_No_of_persons"})
```

Enrich Layer

Add detailed demographic data and statistics to your analysis

```
enriched_crime = enrich_data.enrich_layer(police_beats,
                                          analysis_variables=analysis_variables)
```

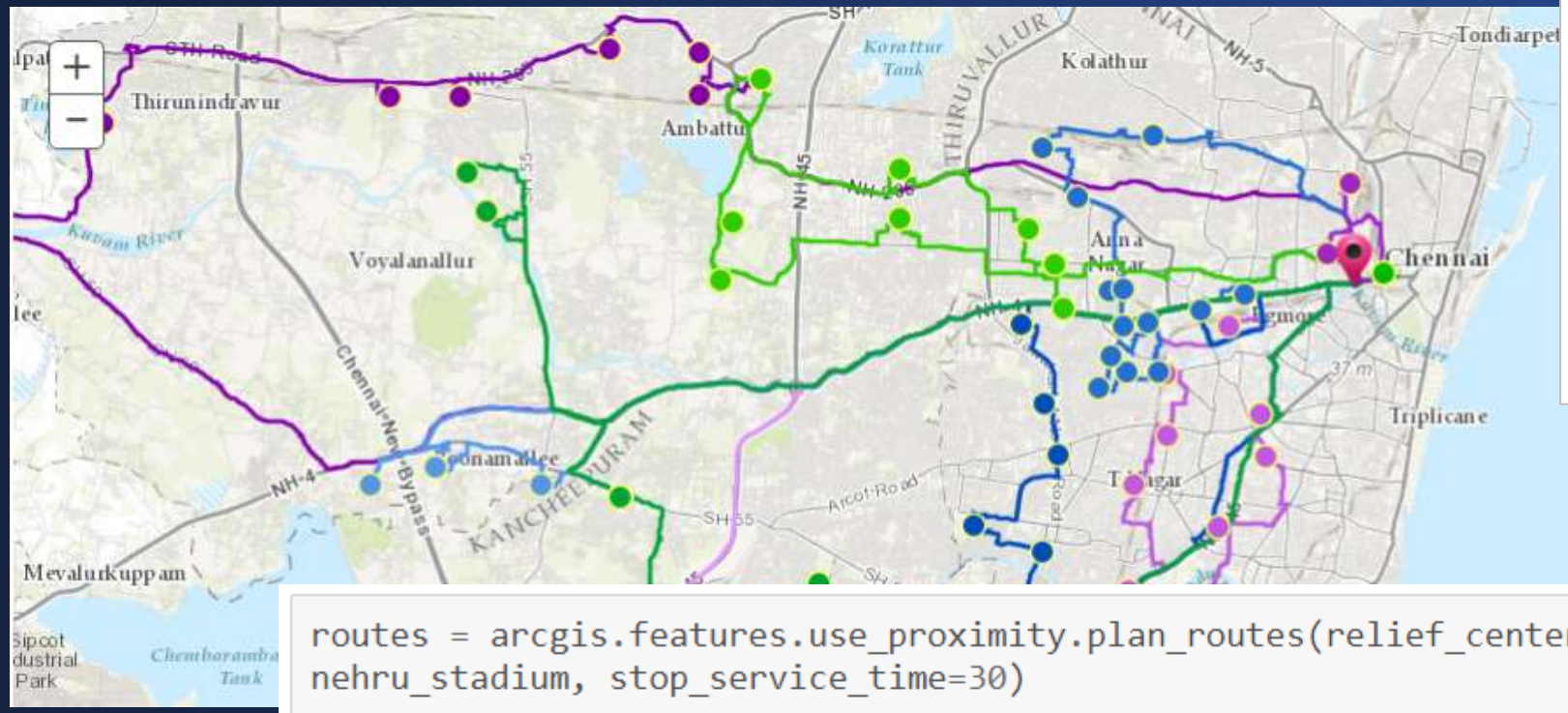
```
Submitted.
Executing...
```

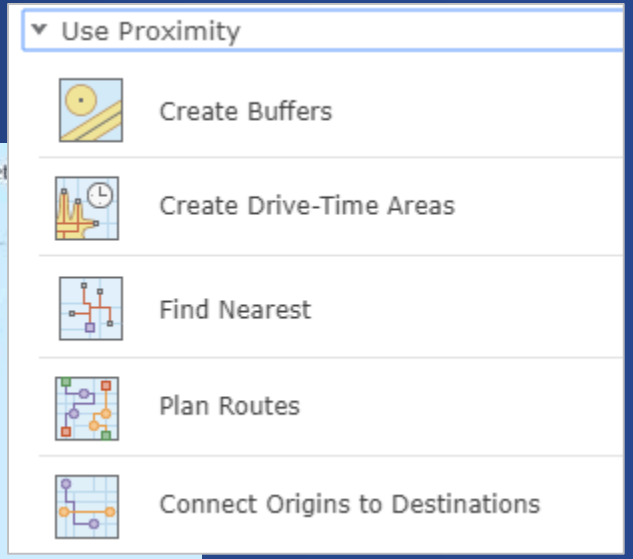
```
enriched_df = enriched_crime.query().df
enriched_df.head()
```

	ASSCDEG_CY	AVGFMSZ_CY	AVGHHSZ_CY	AVGHINC_CY	BACHDEG_CY	DIVINDX_CY	EDUCBASECY	ENRICH_FID
0	0.0	0.00	0.00	0.0	0.0	0.0	0.0	1.0
1	0.0	3.57	2.29	93321.0	16.0	79.6	30.0	2.0
2	545.0	2.34	1.35	99836.0	1142.0	72.8	12007.0	3.0
3	869.0	2.55	1.64	112053.0	8333.0	63.8	20489.0	4.0

Use Proximity

"What is near what?"





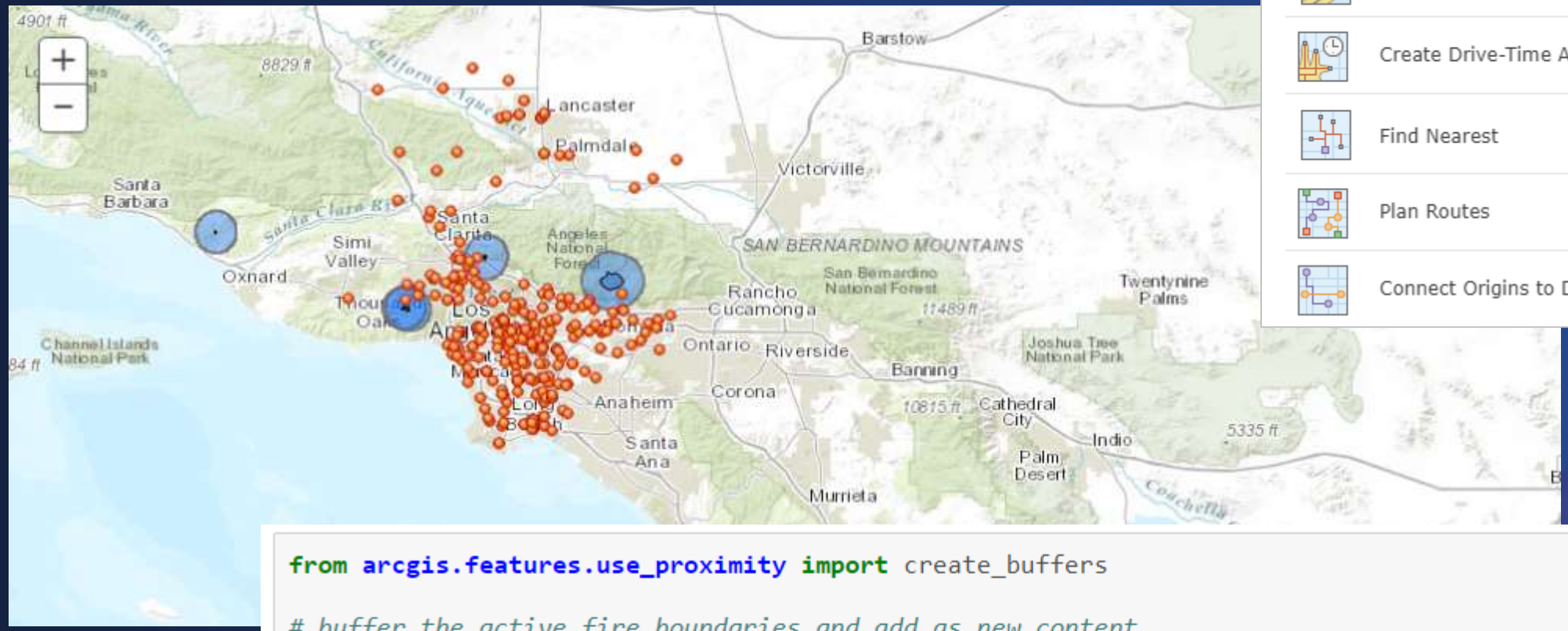

```
routes = arcgis.features.use_proximity.plan_routes(relief_centers, 15, 15, start_time,
nehru_stadium, stop_service_time=30)
```

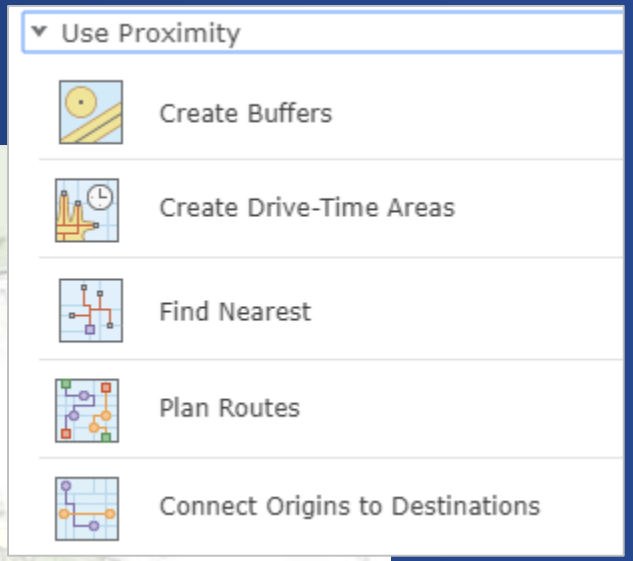
```
routemap.add_layer(routes['routes_layer'])
```

```
routemap.add_layer(routes['assigned_stops_layer'])
```

Use Proximity

"What is near what?"

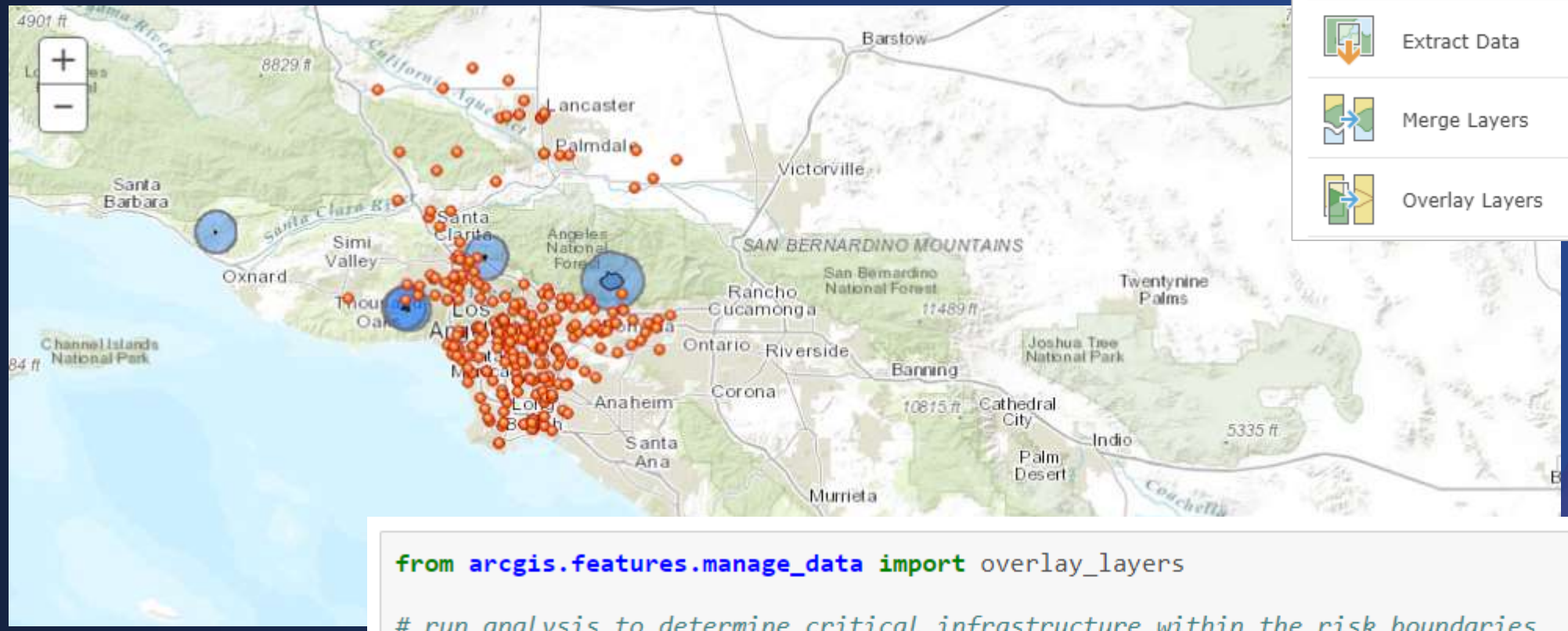


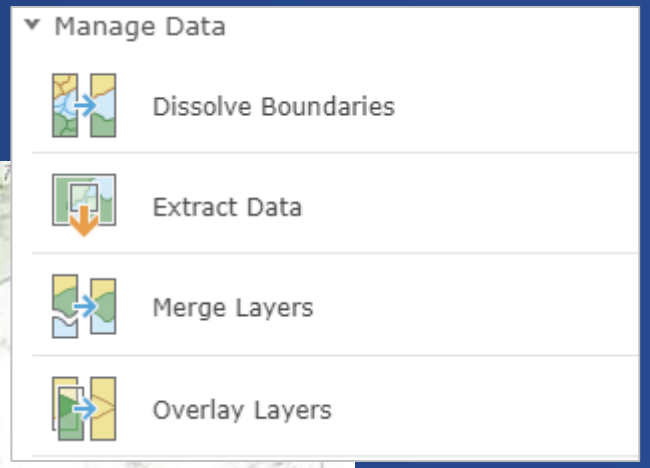



```
from arcgis.features.use_proximity import create_buffers

# buffer the active fire boundaries and add as new content

timestamp = '{:%Y_%m_%d_%H_%M_%S}'.format(datetime.datetime.now())
firebuffers = create_buffers(fires, [4], None, 'Miles', output_name="Fire_Buffers_" + timestamp
)
```

Manage Data

Manage geographic data, overlay layers

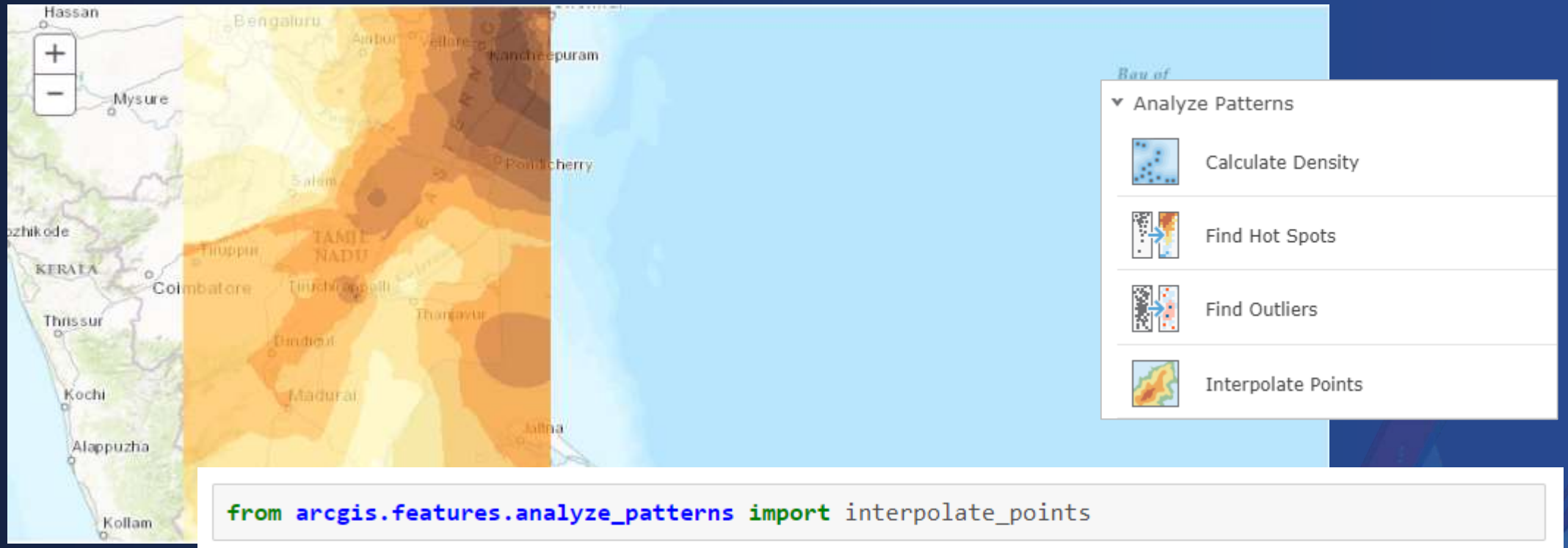





```
from arcgis.features.manage_data import overlay_layers

# run analysis to determine critical infrastructure within the risk boundaries
riskinfra = overlay_layers(firebuffers, infra,

                           overlay_type="Intersect",
                           output_name="At_Risk_Infrastructure_" + timestamp)
```

Analyze Patterns

Identify, quantify, and visualize spatial patterns in your data.




```
from arcgis.features.analyze_patterns import interpolate_points
```

```
interpolated_rf = interpolate_points(rainfall, field='RAINFALL')
```

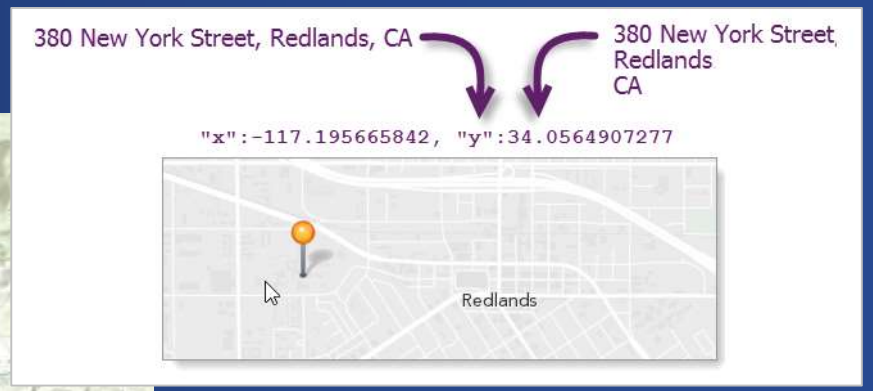
Geocoding

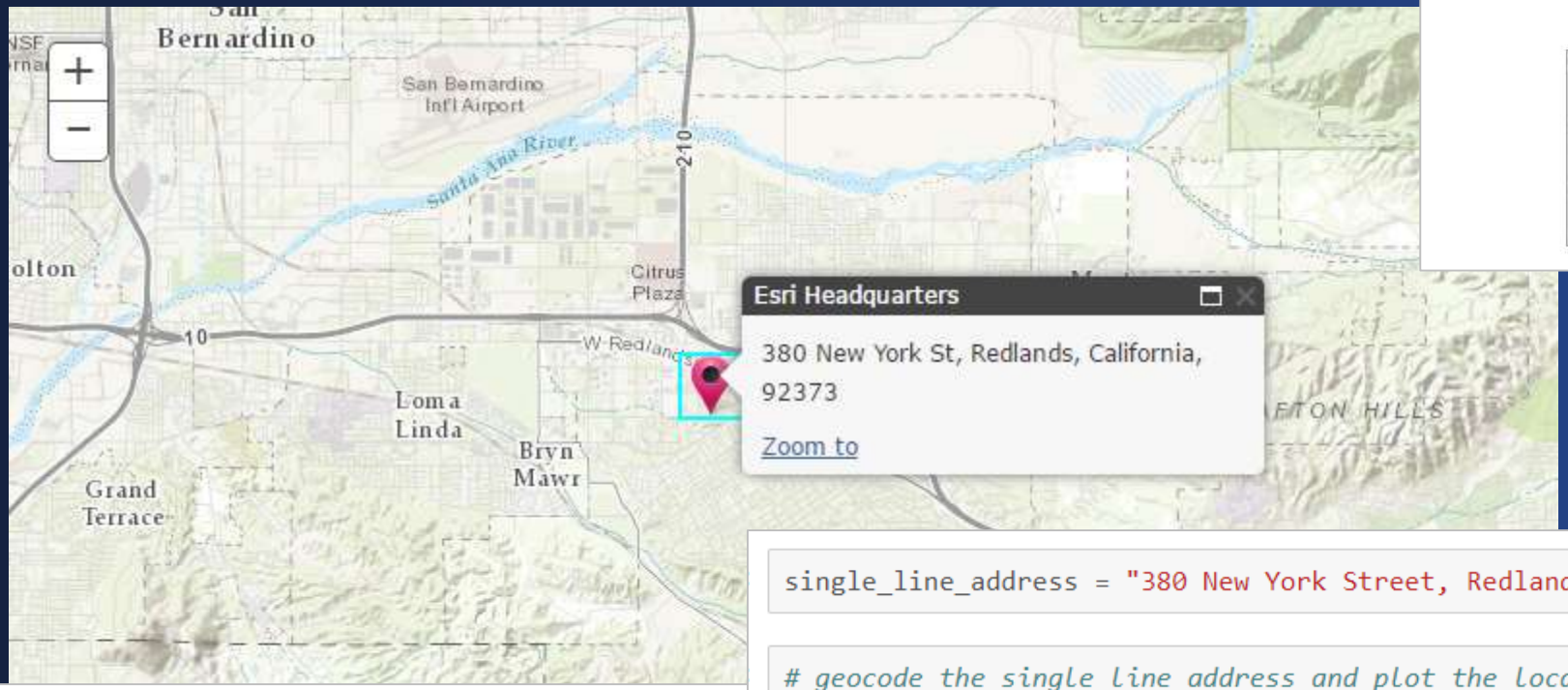
Geocoding, Batch geocoding, reverse geocoding

`arcgis.geocoding` module

Geocode

Single line or multi field addresses






```
single_line_address = "380 New York Street, Redlands, CA 92373"
```

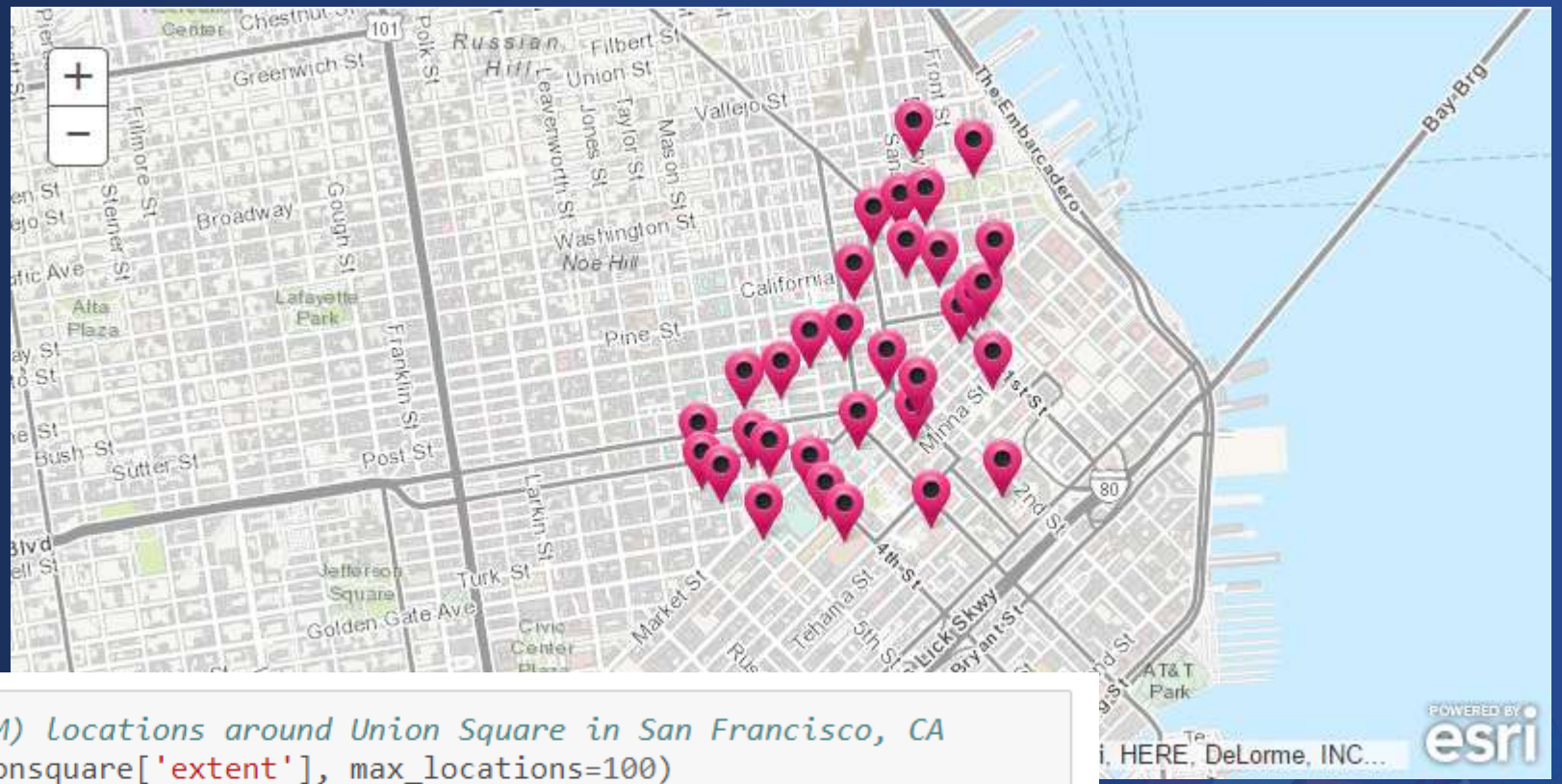
```
# geocode the single line address and plot the location of the first geocode result on the map
esrihq = geocode(single_line_address)[0]

# add a popup to the matched location
popup = {
    "title" : "Esri Headquarters",
    "content" : esrihq['address']
    }
map.draw(esrihq['location'], popup)
```

```
multi_field_address = {
    "Address" : "380 New York Street",
    "City" : "Redlands",
    "Region" : "CA",
    "Postal" : 92373
    }
```

Geocode

Points of Interest

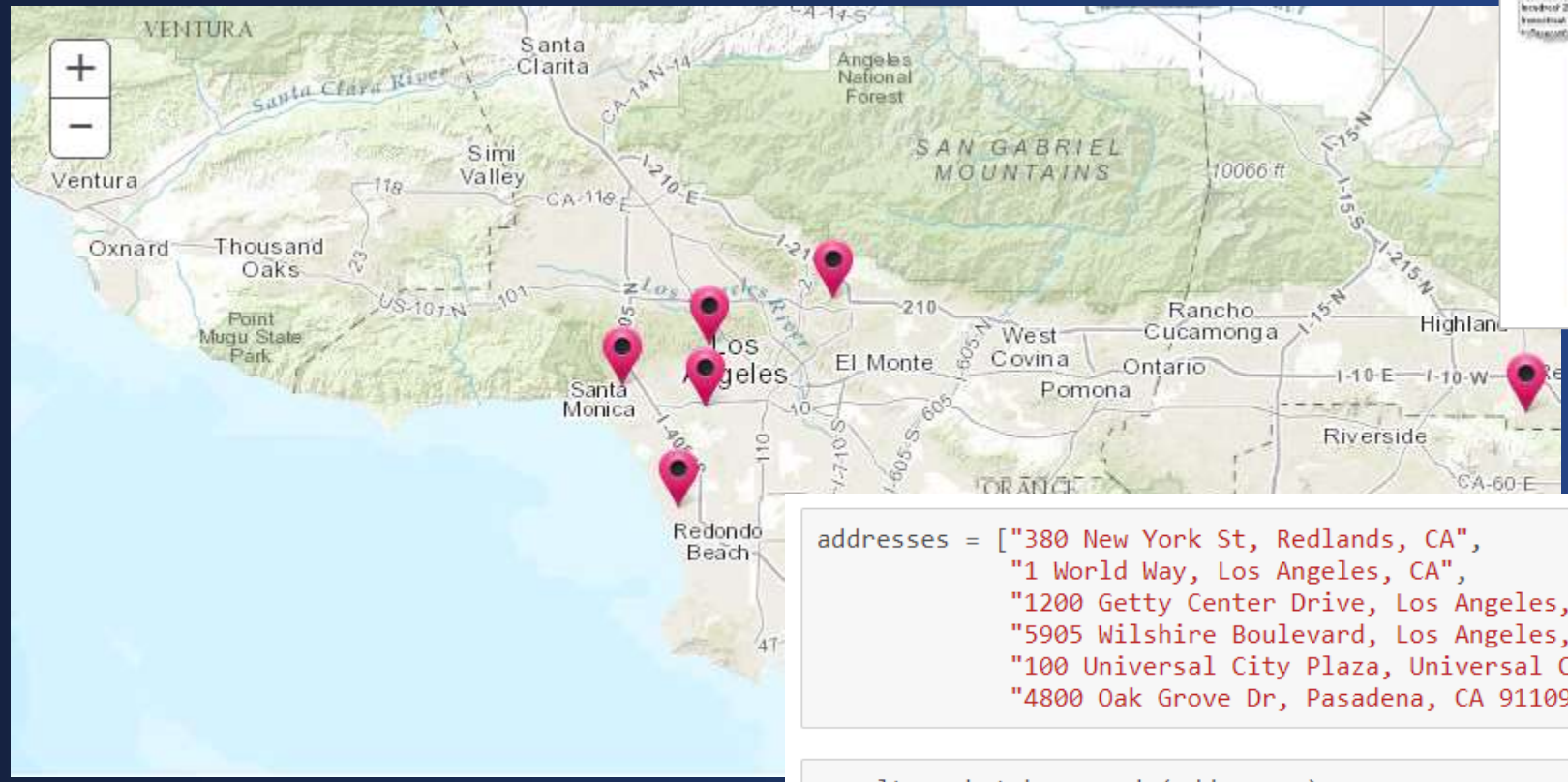
- Cultural or geographic landmarks
- Businesses by name or category
- Administrative divisions




```
# find and plot upto 100 Starbucks(TM) locations around Union Square in San Francisco, CA
starbucks = geocode("Starbucks", unionsquare['extent'], max_locations=100)
for starbuck in starbucks:
    map.draw(starbuck['location'])
```

Batch Geocoding

Geocode an entire list of single line or multi field addresses




```
addresses = ["380 New York St, Redlands, CA",
             "1 World Way, Los Angeles, CA",
             "1200 Getty Center Drive, Los Angeles, CA",
             "5905 Wilshire Boulevard, Los Angeles, CA",
             "100 Universal City Plaza, Universal City, CA 91608",
             "4800 Oak Grove Dr, Pasadena, CA 91109"]
```

```
results = batch_geocode(addresses)
```

```
for address in results:
    map.draw(address['location'])
```

Reverse Geocode

Determines address at a particular x/y location


```
result = reverse_geocode([4.366281,50.851994], lang_code="fr")
```

```
result
```

```
{'address': {'Address': 'Rue de la Sablonnière 15',
  'City': 'Bruxelles',
  'CountryCode': 'BEL',
  'Loc_name': 'BEL.PointAddress',
  'Match_addr': 'Rue de la Sablonnière 15, 1000, Bruxelles',
  'Neighborhood': 'Bruxelles',
  'Postal': '1000',
  'PostalExt': None,
  'Region': 'Bruxelles',
  'Subregion': 'Bruxelles'},
 'location': {'spatialReference': {'latestWkid': 4326, 'wkid': 4326},
  'x': 4.366265813154625,
  'y': 50.85196404988331}}
```

Network Analysis

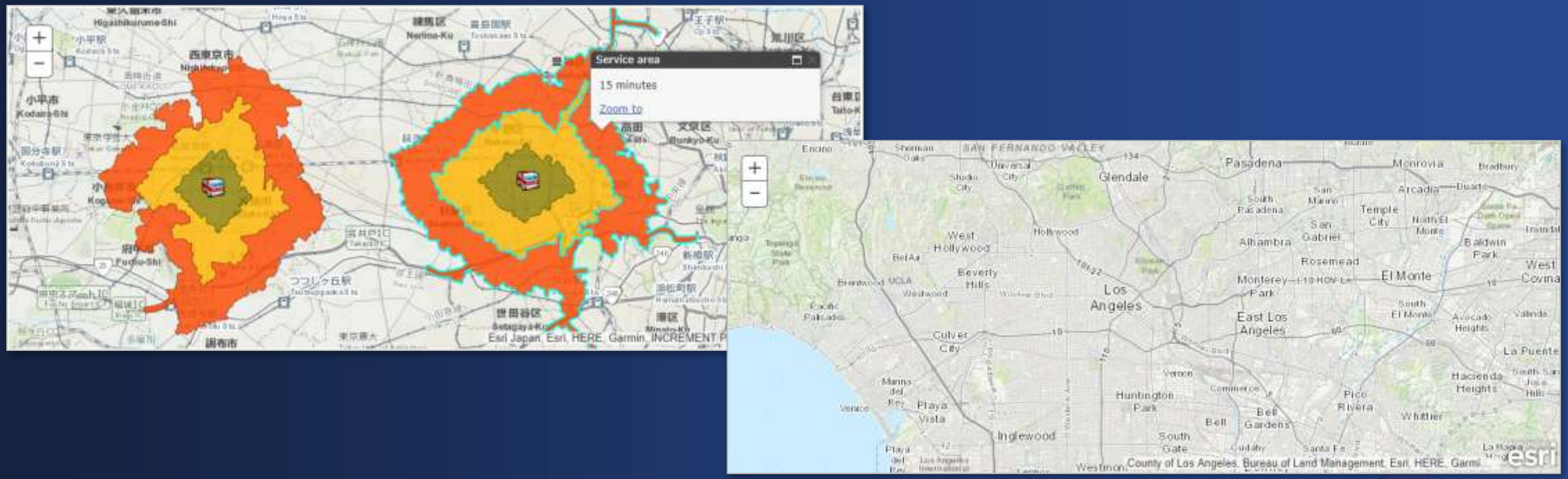
Routing and directions, location allocation, service areas…

`arcgis.network.analysis` module

Network analysis capabilities and tools

Commercial grade, traffic aware routing and directions for multiple travel modes

Network analysis – Service Areas




```
travel_modes = sa_layer.retrieve_travel_modes()
truck_mode = [t for t in travel_modes['supportedTravelModes'] if t['name'] == 'Trucking Time']
[0]

result = sa_layer.solve_service_area(fire_station_fset, default_breaks=[5,10,15],
                                     travel_direction='esriNATravelDirectionFromFacility',
                                     travel_mode=truck_mode)
```

GeoEnrichment

Enrich your analysis with demographic and business data

`arcgis.geoenrichment` module

GeoEnrichment

- Get facts about a location or area
  - Street addresses
  - Points, lines and polygon geometries
  - Within a drive time or service area
  - Named geographical areas
    - Counties or block groups in California
    - Districts and subdistricts in India
- Create charts and choropleth maps

```
df = usa.data_collections

# print a few rows of the DataFrame
df.head()
```

dataCollectionID	analysisVariable	alias	fieldCategory	vintage
1yearincrements	1yearincrements.AGE0_CY	2017 Population Age <1	2017 Age: 1 Year Increments (Esri)	2017
1yearincrements	1yearincrements.AGE1_CY	2017 Population Age 1	2017 Age: 1 Year Increments (Esri)	2017
1yearincrements	1yearincrements.AGE2_CY	2017 Population Age 2	2017 Age: 1 Year Increments (Esri)	2017

```
ca_counties = usa.subgeographies.states['California'].counties
```

```
counties_df = enrich(study_areas=ca_counties, data_collections=['Age'])
counties_df.head(10)
```

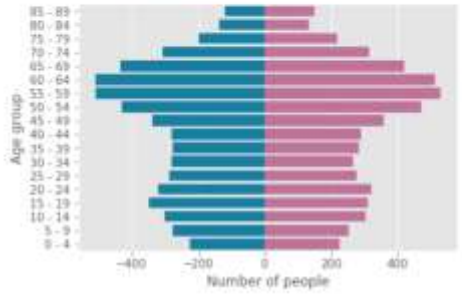
	FEM0	FEM10	FEM15	FEM20	FEM25	FEM30	FEM35	FEM40	FEM45	FEM5	...	MALE75	MALE8
0	47323	50068	50590	57125	60205	61600	59673	56910	56069	49093	...	16318	10422
1	24	33	29	20	24	24	23	25	34	33	...	17	10
2	706	871	867	725	747	790	809	821	1019	769	...	789	510

Age Pyramid of affected population

```
print('Number of people in fire perimeter: ' + str(popdf['female'].sum() - popdf['male'].sum()))
```

```
Number of people in fire perimeter: 11228
```

```
sns.barplot(x="female", y="agelabel", color="#CC6699", label="Female", data=popdf, edgecolor='none')
sns.barplot(x="male",   y="agelabel", color="#008AB8", label="Male",   data=popdf,  edgecolor='none')
plt.ylabel('Age group')
plt.xlabel('Number of people');
```



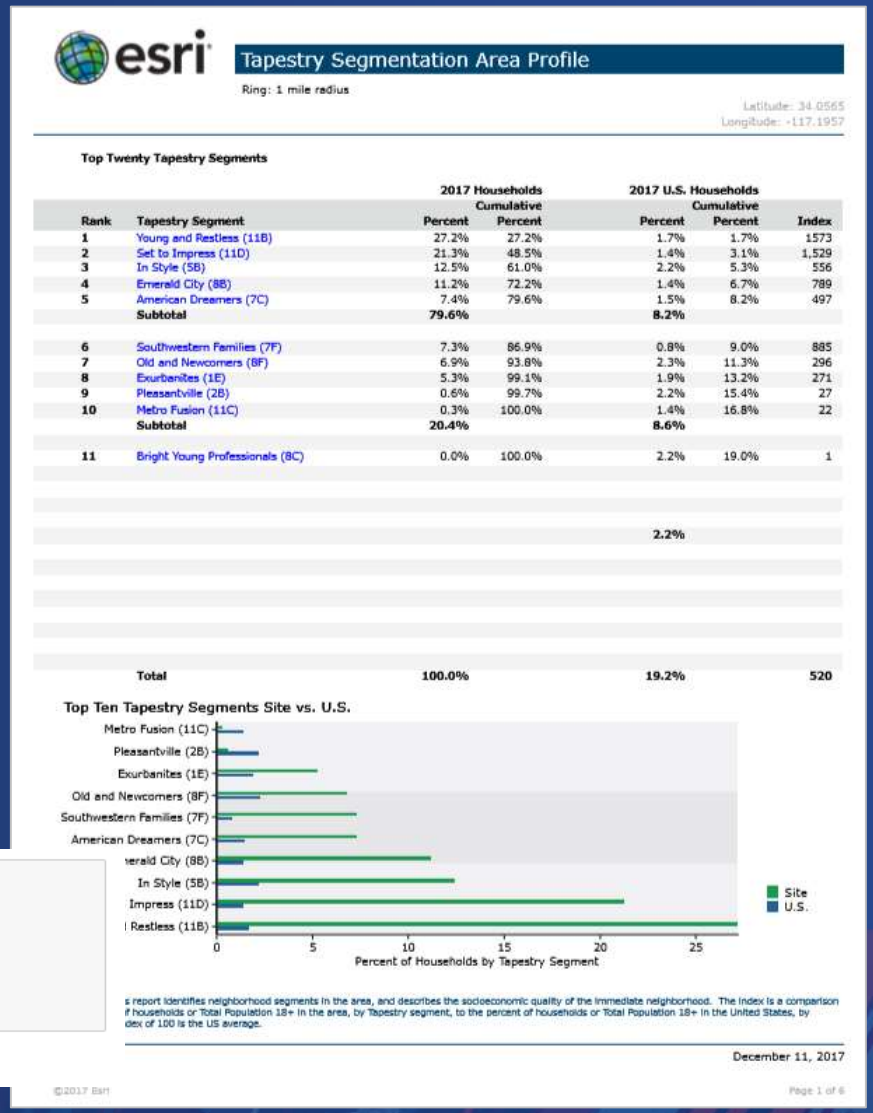
GeoEnrichment – create reports

```
# print a sample of the reports available for USA
usa.reports.head(10)
```

	id	title	categories	formats
0	census2010_profile	2010 Census Profile	[Demographics]	[pdf, xlsx]
1	acs_housing	ACS Housing Summary	[Demographics]	[pdf, xlsx]
2	acs_population	ACS Population Summary	[Demographics]	[pdf, xlsx]
3	55plus	Age 50+ Profile	[Demographics]	[pdf, xlsx]
4	agesexrace	Age by Sex by Race Profile	[Demographics]	[pdf, xlsx]
5	agesex	Age by Sex Profile	[Demographics]	[pdf, xlsx]
6	cex_auto	Automotive Aftermarket Expenditures	[Consumer Spending]	[pdf, xlsx]
7	business_loc	Business Locator	[Business]	[pdf, xlsx]
8	business_summary	Business Summary	[Business]	[pdf, xlsx]
9	community_profile	Community Profile	[Demographics]	[pdf, xlsx]

```
report = create_report(study_areas=["380 New York Street, Redlands, CA"],
                       report="tapestry_profileNEW",
                       export_format="PDF",
                       out_folder=r"c:\xc", out_name="esri_tapestry_profile.pdf")
report
```

```
'c:\\xc\\esri_tapestry_profile.pdf'
```



esri

Tapestry Segmentation Area Profile

Ring: 1 mile radius

Latitude: 34.0565
Longitude: -117.1957

Top Twenty Tapestry Segments

Rank	Tapestry Segment	2017 Households Percent	2017 Households Cumulative Percent	2017 U.S. Households Percent	2017 U.S. Households Cumulative Percent	Index
1	Young and Restless (11B)	27.2%	27.2%	1.7%	1.7%	1573
2	Set to Impress (11D)	21.3%	48.5%	1.4%	3.1%	1,529
3	In Style (5B)	12.5%	61.0%	2.2%	5.3%	556
4	Emerald City (8B)	11.2%	72.2%	1.4%	6.7%	789
5	American Dreamers (7C)	7.4%	79.6%	1.5%	8.2%	497
	Subtotal	**79.6%**		**8.2%**		
6	Southwestern Families (7F)	7.3%	86.9%	0.8%	9.0%	885
7	Old and Newcomers (8F)	6.9%	93.8%	2.3%	11.3%	296
8	Exurbanites (1E)	5.3%	99.1%	1.9%	13.2%	271
9	Pleasantville (2B)	0.6%	99.7%	2.2%	15.4%	27
10	Metro Fusion (11C)	0.3%	100.0%	1.4%	16.8%	22
	Subtotal	**20.4%**		**8.6%**		
11	Bright Young Professionals (8C)	0.0%	100.0%	2.2%	19.0%	1
				2.2%		
	Total	**100.0%**		**19.2%**		**520**

Top Ten Tapestry Segments Site vs. U.S.

s report identifies neighborhood segments in the area, and describes the socioeconomic quality of the immediate neighborhood. The index is a comparison f households or Total Population 18+ in the area, by Tapestry segment, to the percent of households or Total Population 18+ in the United States, by dex of 100 is the US average.

December 11, 2017

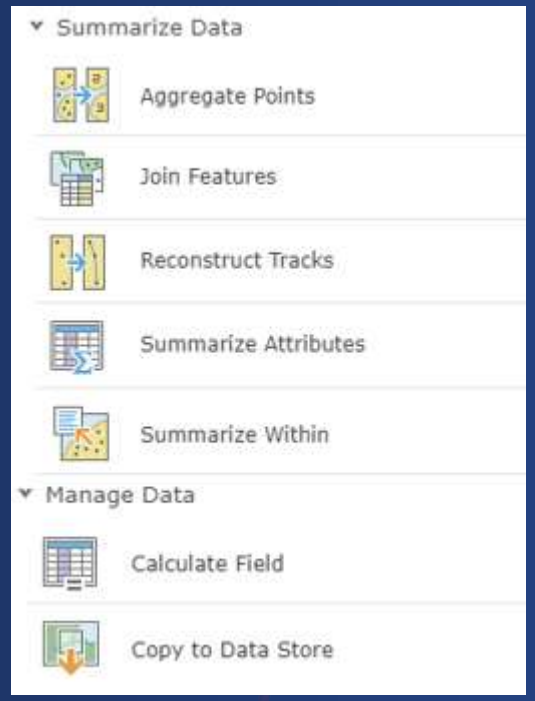
GeoAnalytics

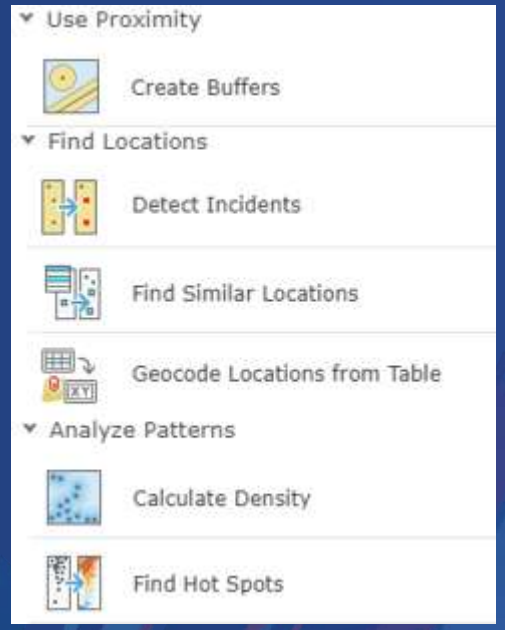
Fast distributed spatio-temporal analysis of large vector and tabular data

`arcgis.geoanalytics` module

GeoAnalytics Tools

- Summarize Data
- Manage Data
- Use Proximity
- Find Locations
- Analyze Patterns





Demo

Analysis of crime patterns in Houston

Attach data

In [27]:
```
datastores = arcgis.geoanalytics.get_datastores()
datastores.add_bigdata('Houston_crime_yearly',
                       r'\\teton\atma_shared\datasets\HoustonCrime')
```

```
Big Data file share exists for Houston_crime_yearly
```

Out[27]:
```
<Datastore title:"/bigDataFileShares/Houston_crime_yearly" type:"bigDataFileSha
re">
```

In [28]:
```
houston_yearly = houston_gis.content.search('Houston_crime_yearly',
                                            'big data file share')[0]
houston_yearly
```

Out[28]:

bigDataFileShares_Houston_crime_yearly
Big Data File Share by admin
Last Modified: March 03, 2017
0 comments, 0 views

In [30]:
```
houston_yearly.layers
```

Out[30]:
```
[<Layer url:"https://dev003247.esri.com/gax/rest/services/DataStoreCatalogs/big
DataFileShares_Houston_crime_yearly/BigDataCatalogServer/houstoncrime2010">,
 <Layer url:"https://dev003247.esri.com/gax/rest/services/DataStoreCatalogs/big
DataFileShares_Houston_crime_yearly/BigDataCatalogServer/houstoncrime2011">,
 <Layer url:"https://dev003247.esri.com/gax/rest/services/DataStoreCatalogs/big
```

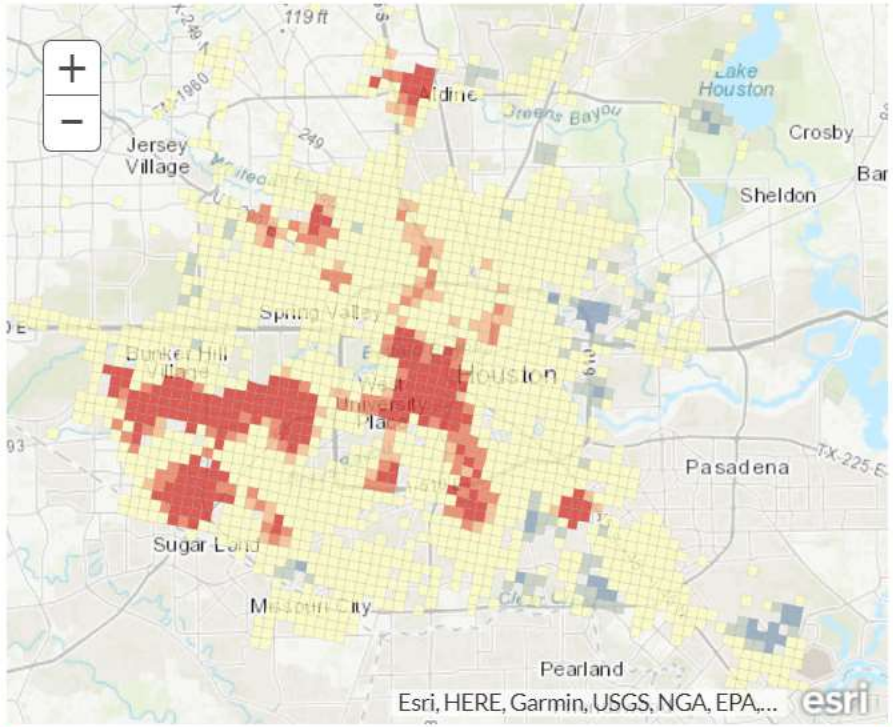
Invoke batch analytics

In [ ]:

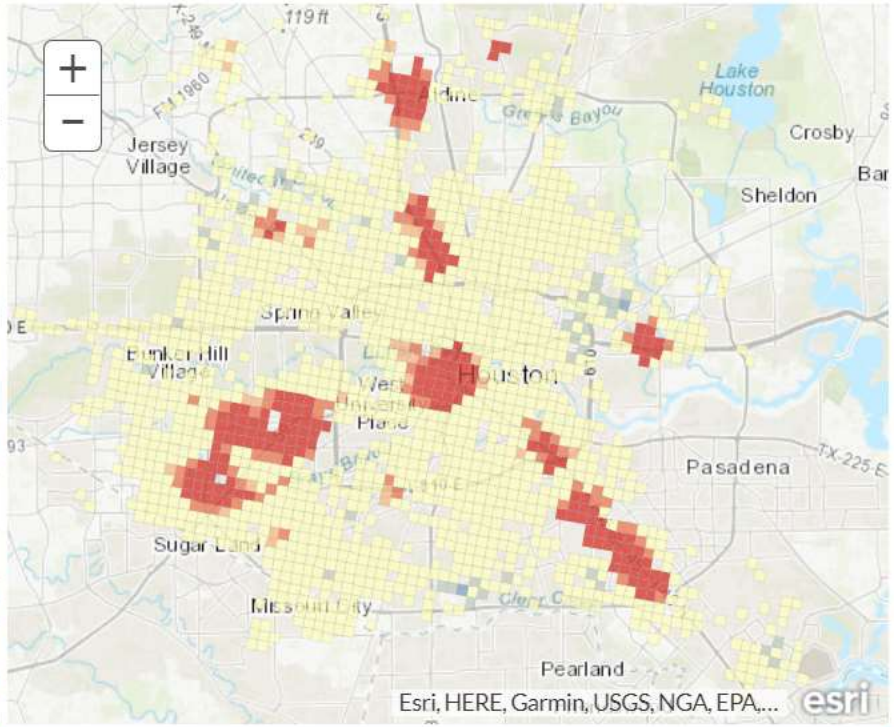
```
for category in df.Category.unique()[:-1]:
    lyrid = 0
    for year in range(2010, 2017):
        output_name='Houston_' + category.replace(' ', '_') + '_Hotspot_' + str
        print('Generating ' + output_name)
        layer = houston_yearly.layers[lyrid]
        layer.filter = "Category='{}'".format(category)

        find_hot_spots(layer, bin_size=0.5, bin_size_unit='Miles',
                       neighborhood_distance=1, neighborhood_distance_unit='Mi
                       output_name=output_name)

        lyrid = lyrid + 1
```

View results
Hot Spots across crime categories

In [33]:
```
display(HBox([hotmap1, hotmap2]))
display(HBox(children=[Button(description='Burglary hot spots in 2016', layout=
                       Button(description='Auto theft hot spots in 2016', layou
            layout=Layout(width='100%')))
```



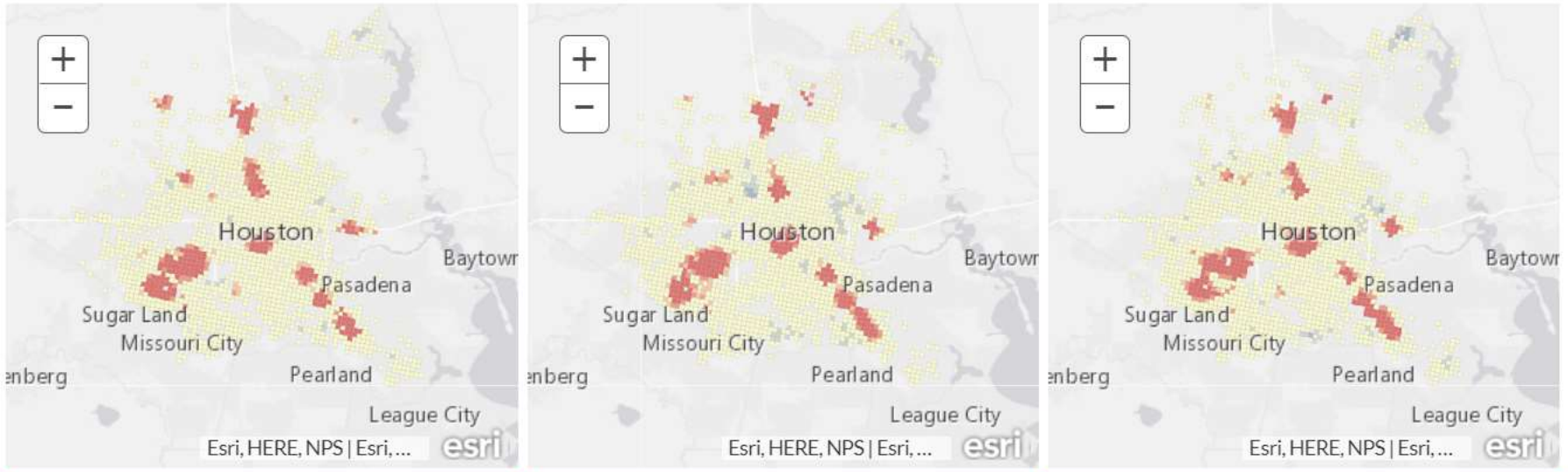
Burglary hot spots in 2016



Auto theft hot spots in 2016

Compare Hot Spots over time

In [35]:
```
for year in range(2014, 2017):
    layer = houston_gis.content.search('Houston_Auto_Theft_Hotspot_' + str(year
    hotspotmap = houston_gis.map(houston)
    hotspotmap.add_layer(layer)
    hotspotmap.layout=Layout(flex='1 1', padding='3px')
    maps.append(hotspotmap)
    hotspotmap.basemap='gray'
    labels.append(Button(description='Auto theft hot spots in ' + str(year),
                         layout=items_layout, button_style='danger'))

display(HBox([maps[0], maps[1], maps[2]], layout=layout))
display(HBox(children=labels, layout=Layout(width='100%')))
```



Auto theft hot spots in 2014 | Auto theft hot spots in 2015 | Auto theft hot spots in 2016

Imagery and Raster Analysis

On-the-fly and distributed batch analysis of raster data

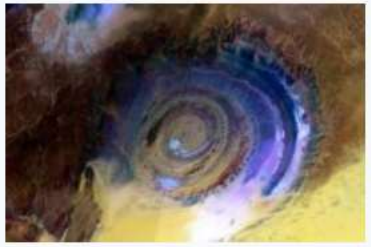
`arcgis.raster` module

Imagery and Raster Analysis

In [3]:
```
landsat_item = gis.content.search('title:Multispectral Landsat',
                                  'Imagery Layer', outside_org=True)[0]
```

In [4]:
```
landsat_item
```

Out[4]:



Multispectral Landsat

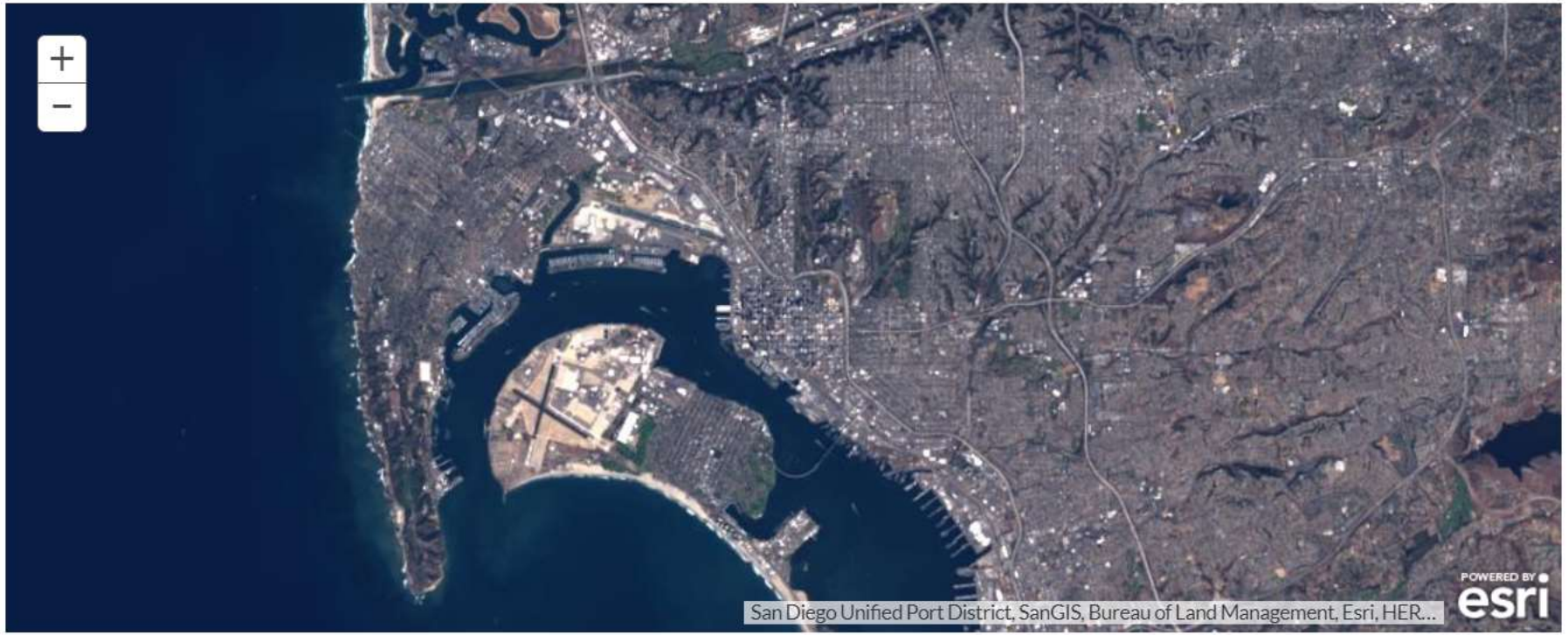
Landsat 8 OLI, 30m Multispectral 8 band scenes with visual renderings and indices. Updated daily. Based on the Landsat on AWS collections.

Imagery Layer by esri
Last Modified: December 07, 2016
0 comments, 118,273 views

In [5]:
```
landsat = landsat_item.layers[0]
```

Visualizing imagery layers

In [6]:
```
imagery_map = gis.map('San Diego, CA', zoomlevel=12)
imagery_map.add_layer(landsat)
imagery_map
```



In [27]:
```
for rasterfunc in landsat.properties.rasterFunctionInfos:
    print(rasterfunc.name)
    imagery_map.add_layer(apply(landsat, rasterfunc.name))
    time.sleep(2)
```

```
Agriculture with DRA
Bathymetric with DRA
```

Custom raster processing using Raster Functions

Running in San Diego

Least Important

20%

Low Elevation

Moderately Important

30%

Flat, Not Hilly

Most Important

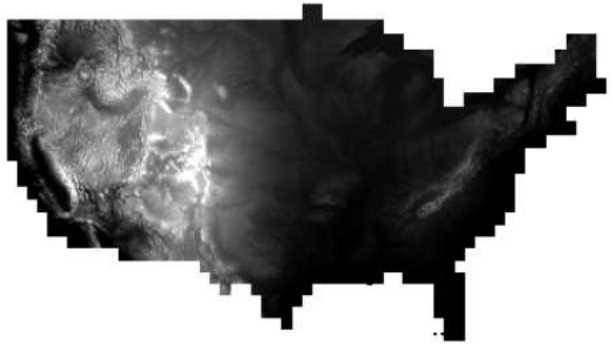
50%

Natural, Not Built

Inputs - Elevation

In [8]:
```
# Digital elevation model for the US

elevation_item = enterprise.content.search('elevation_270m')[0]
elevation_lyr = elevation_item.layers[0]
elevation_lyr
```

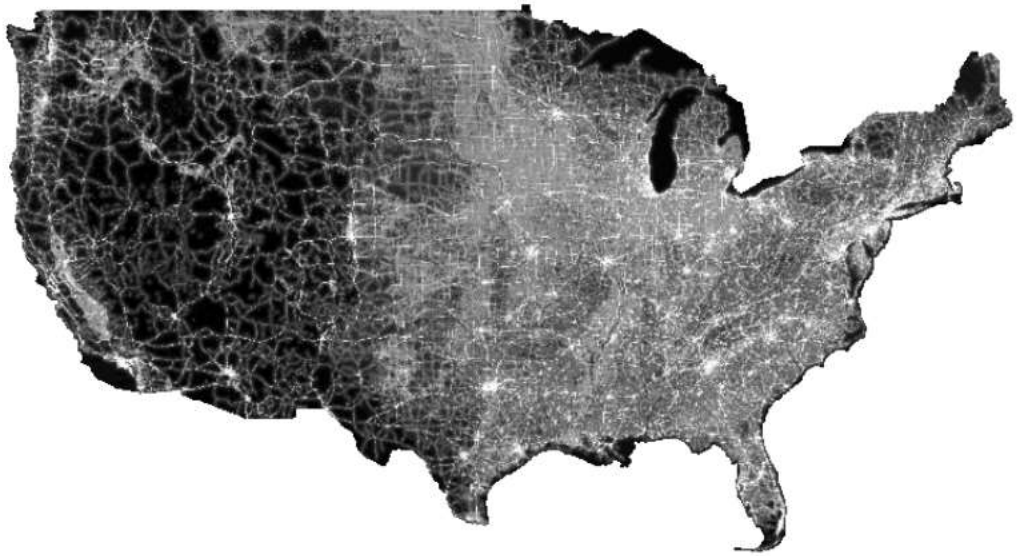
Out[8]:



Natural areas

In [9]:
```
# Human Modified Index imagery layer
# This dataset is based on research on the degree of human modification to
# the landscape, on a scale of 0 - 1, where 0.0 indicates unmodified natural
# landscape and 1.0 indicates the landscape is completely modified by human ac

naturalareas_item = enterprise.content.search('human_modification_index')[0]
naturalareas_lyr = naturalareas_item.layers[0]
naturalareas_lyr
```

Out[9]:



Interactive raster processing in Jupyter Notebook

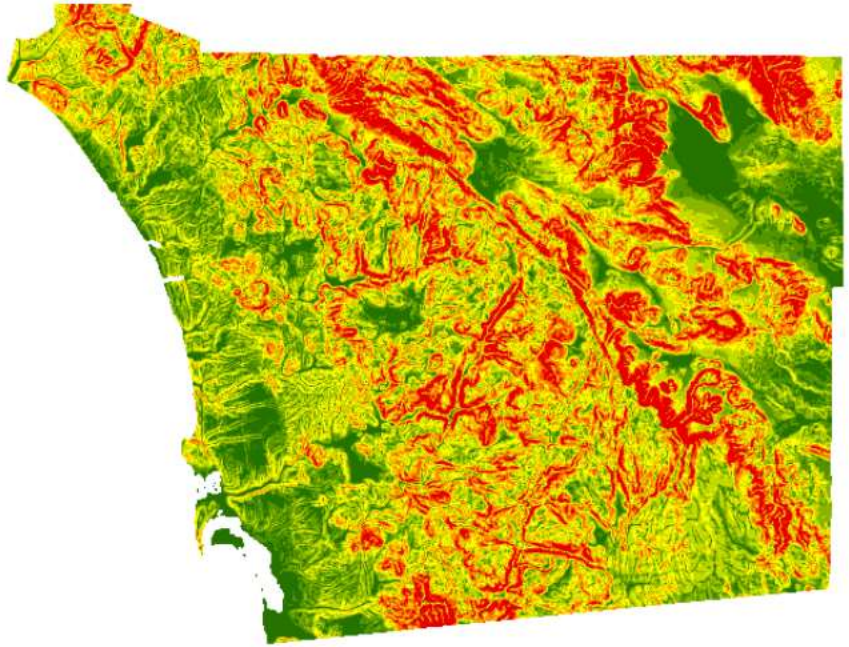
In [12]:
```
clipped_elev = clip(elevation_lyr, sd_geom)
clipped_elev
```

Out[12]:

Chaining raster functions

In [14]:
```
output_values = [1,2,3,4,5,6,7,8,9]

colormap(remap(slope(clipped_elev,
                     slope_type='DEGREE',
                     z_factor=1),
               input_ranges=[0,1, 1,2, 2,3, 3,5, 5,7, 7,9, 9,12, 12,15, 15,100],
               output_values=output_values),
         colormap=red_green)
```

Out[14]:



Prepare input layers

In [17]:
```
elevation = remap(elevation_lyr,
                  [-90,250, 250,500, 500,750, 750,1000, 1000,1500, 1500,2000,
                  output_values)
```

In [18]:
```
terrain = remap(slope(elevation_lyr, slope_type='DEGREE', z_factor=1), # Slope
                  [0,1, 1,2, 2,3, 3,5, 5,7, 7,9, 9,12, 12,15, 15,100],
                  output_values)
```

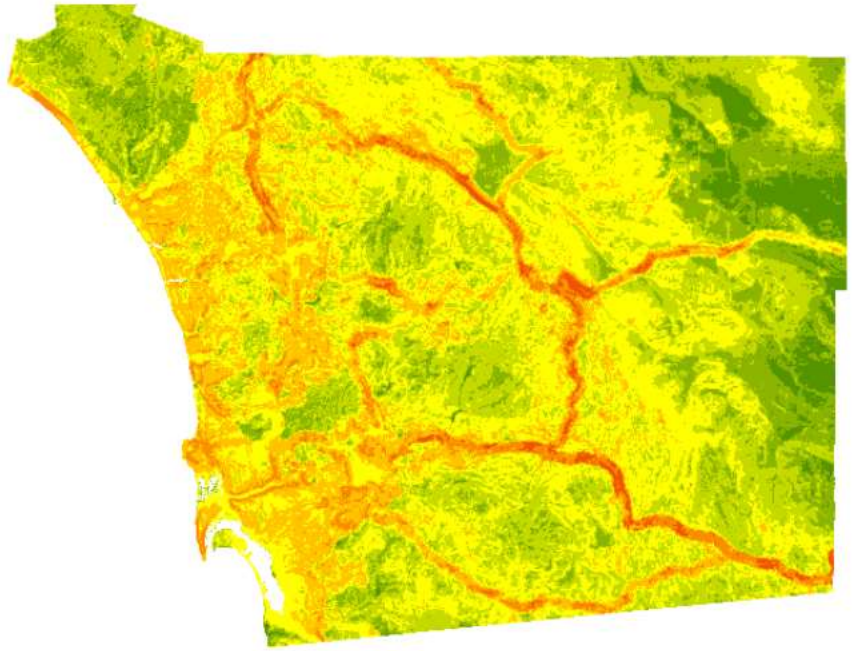
In [19]:
```
natural_areas = remap(naturalareas_lyr,
                  [0.0,0.1, 0.1,0.2, 0.2,0.3, 0.3,0.4, 0.4,0.5,0.5,0.6, 0.6,
                  output_values)
```

Map Algebra for the Web GIS

In [20]:
```
result = 0.2*elevation + 0.3*terrain + 0.5*natural_areas
```

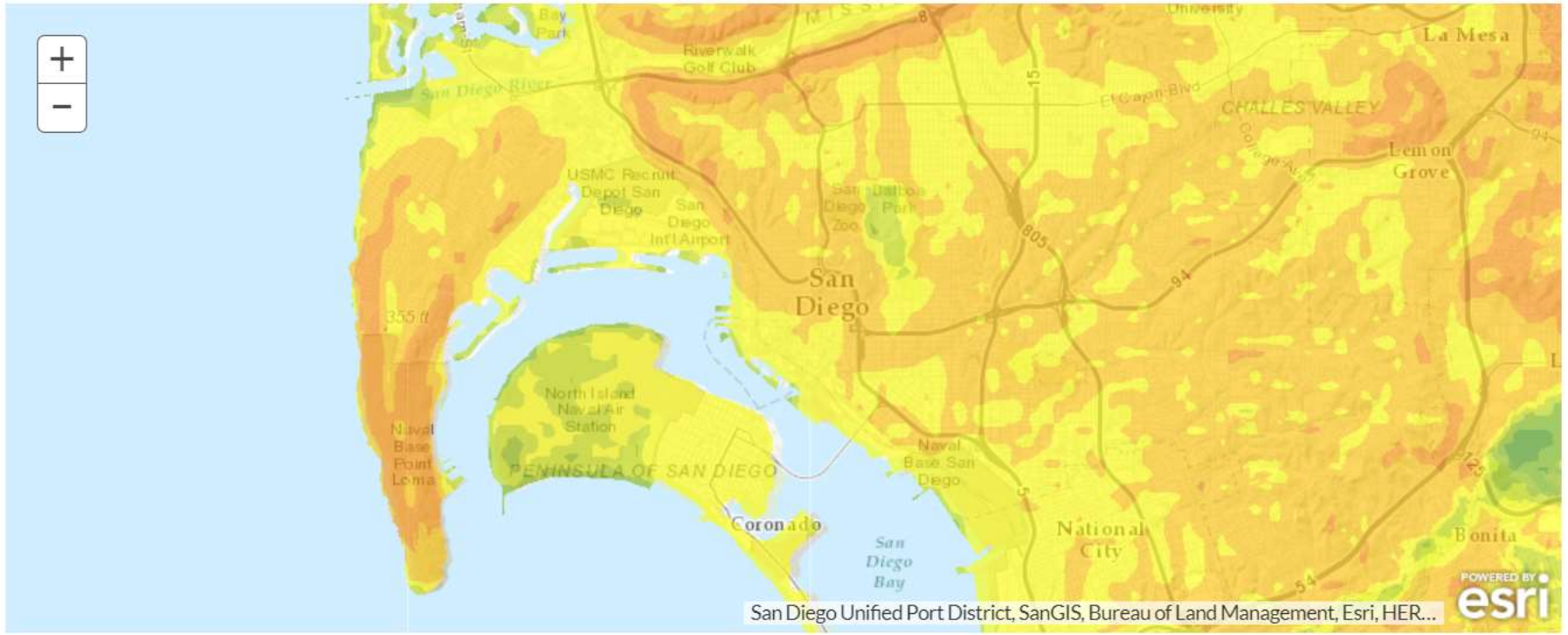
In [21]:
```
run_raster = colormap(clip(result, sd_geom), colormap=red_green)
run_raster
```

Out[21]:



Visualize results using map widget

In [22]:
```
surface_map = gis.map('San Diego, CA', zoomlevel=12)
surface_map
```



In [23]:
```
surface_map.add_layer(run_raster, {'opacity': 0.6})
```

Persist results as an imagery layer

In [24]:
```
# Generate a persistent result at source resolution using Raster Analytics
resultlyr = run_raster.save('SanDiego_PlacesToRun')
```

In [26]:
```
resultlyr
```

Out[26]:

SanDiego_PlacesToRun
Analysis Image Service generated from GenerateRaster
Imagery Layer by arcgis_python
Last Modified: July 06, 2017
0 comments, 0 views

Spatial Data Frame

Feature layers as Pandas' dataframe

`arcgis.features.SpatialDataFrame` class

What is a Spatial Data Frame?

- **Built on Pandas**
- **Easy to Update, Query and Add New Content**
- **Quickly visualize data**

Built to do Work!

- Interoperable with OCG and Esri Geometries
- Consumes multiple Data Sources:
  - Feature Layers
  - Feature Classes
  - CSV
  - Other data frames

Benefits of Spatial Data Frame

- **Packs large datasets into memory efficiently**
- **Spatial Indexing**
- **Consumes multiple data format**
- **Small footprint with BIG results**
- **Using Pandas v.23**

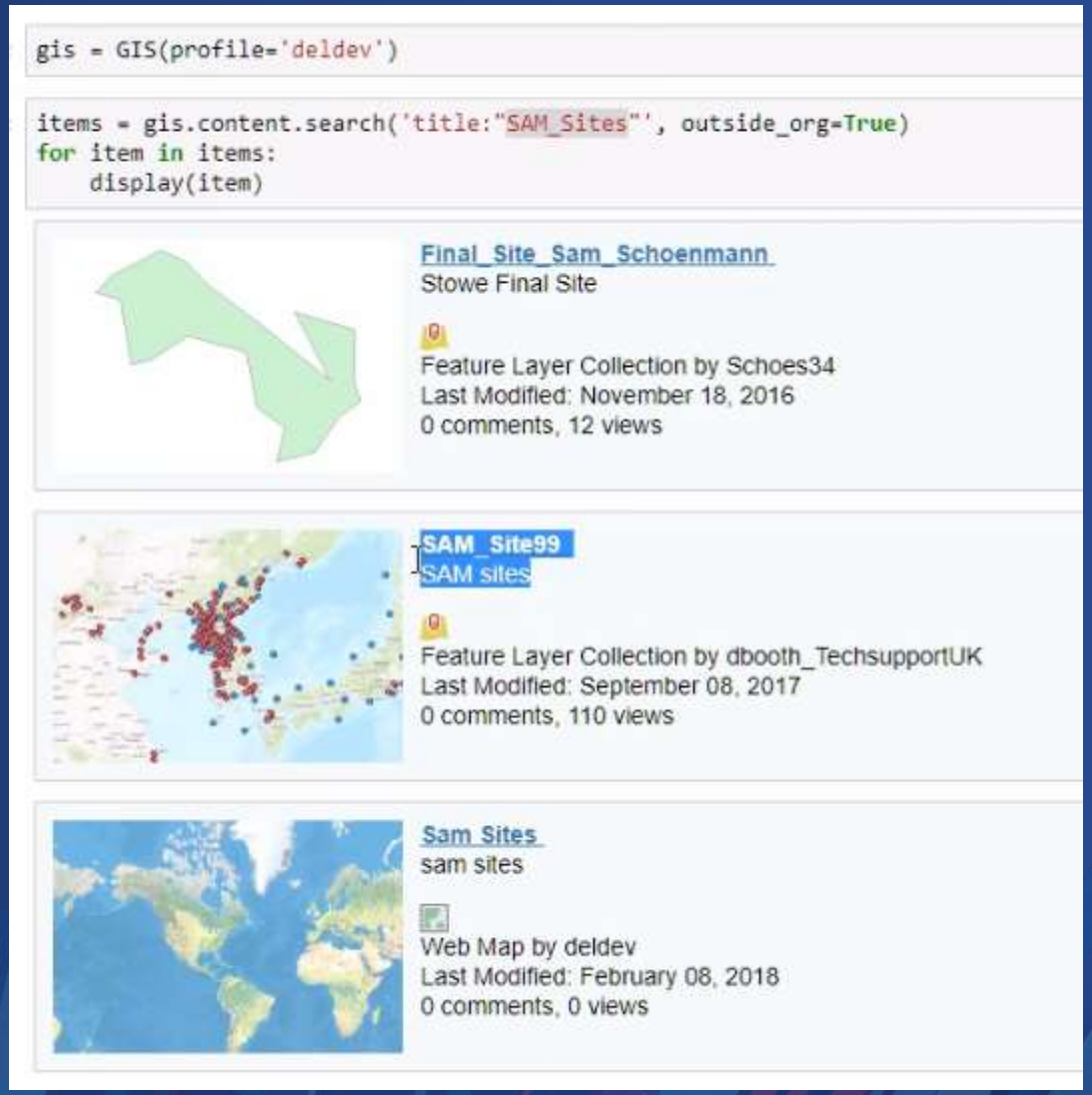
Geometry Engines

- The DataFrame accessor follows the following pattern:
  1. ArcGIS Geometry Engine
  2. Shapely Geometry Engine
  3. Nothing
- Allows for GIS on multiple platforms
  - Linux, Max, Windows

Spatially Enabled Data Frame Demo

Data Science with ArcGIS - Data

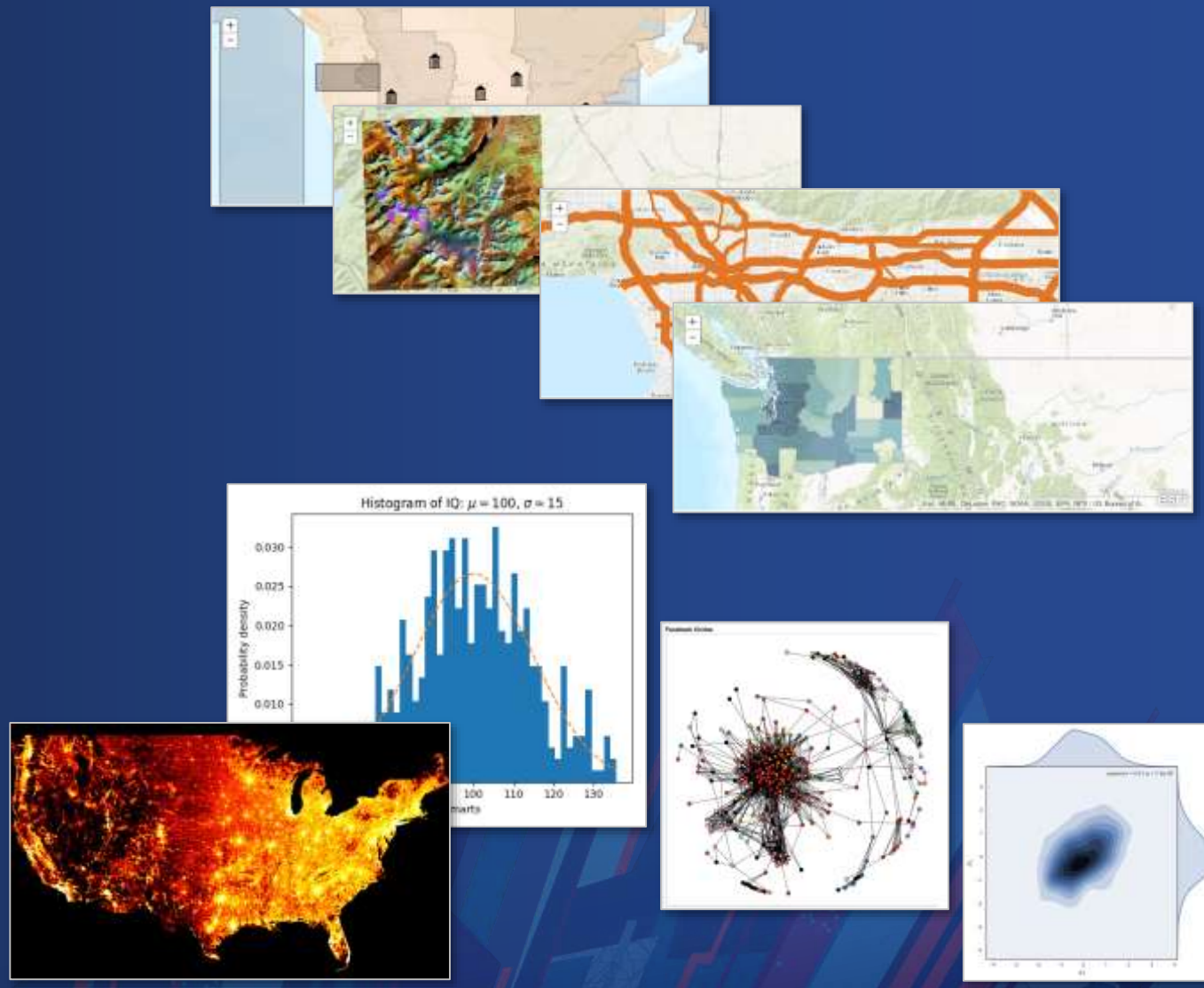
- Esri curated content – Living Atlas
  - Multi-spectral, temporal, dynamic imagery layers
  - Landsat, NAIP, MODIS, Elevation
  - Basemaps, Imagery, Demographics, Transport
  - Boundaries & places, Landscape, Oceans
  - Earth Observations, Urban Systems, Historical Maps, …
- Your data, org's data, data shared with you
  - Shapefiles, File geodatabase, CSV, Excel, HTML, …
  - File shares, cloud share
  - HDFS, Hive and databases
- Public data
  - Maps, layers and datasets shared by users worldwide



Data Science with ArcGIS - Visualization

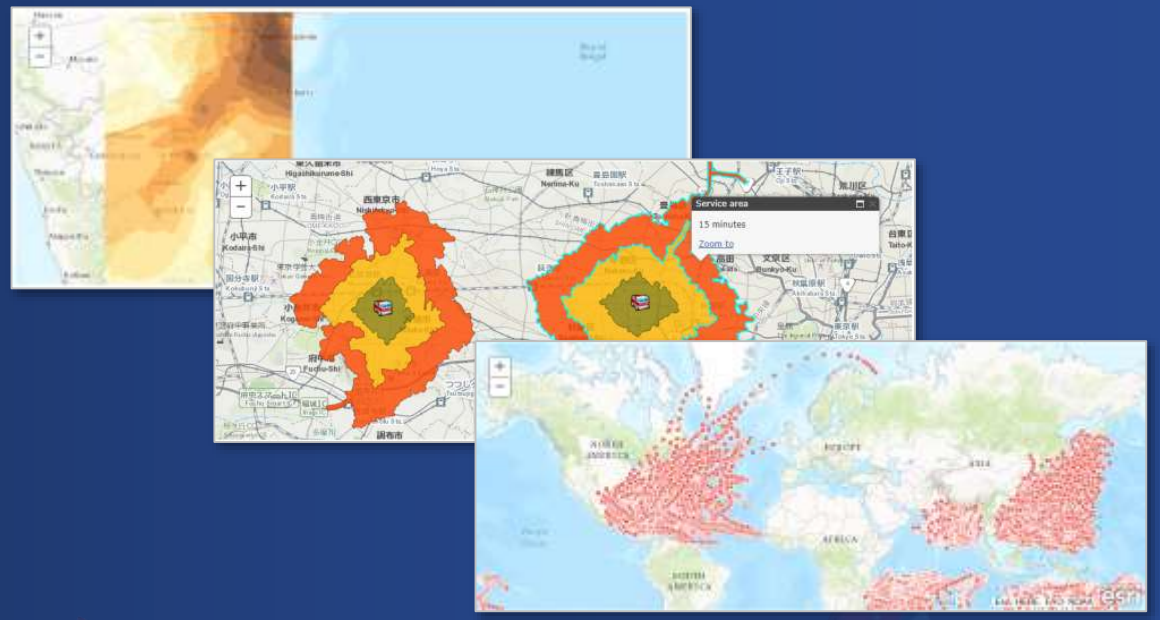
- Visualize with ArcGIS
  - Map widget in Jupyter notebook
  - Web Maps and Web Scene
  - Feature layers
  - Raster and imagery layer
  - Smart mapping
  - Pythonic renderers and symbology

- Visualize with Python
  - Matplotlib, Seaborn, Bokeh, Plotly, …
  - Datashader, Holoviews, Mayavi, …



Data Science with ArcGIS - Analysis

- Analysis with ArcGIS
  - Geoprocessing in Web GIS
    - Spatial analysis, Routing and directions
    - Network analysis, Geocoding, Geoenrichment…
  - Imagery and Raster Analysis
    - On the fly dynamic image processing
    - Distributed raster analysis
  - GeoAnalytics – large tabular and vector data

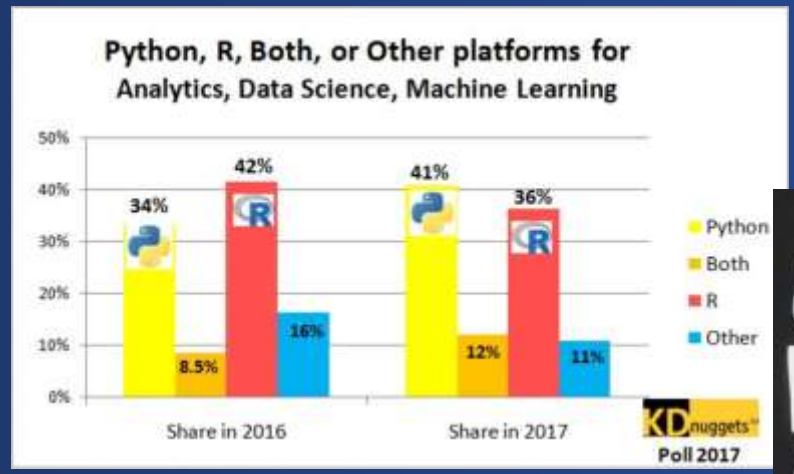


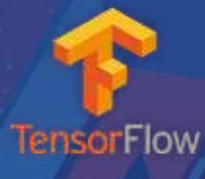
Data Science with ArcGIS - Analysis

- Analysis with Python libraries
  - Data wrangling
    - Pandas, numpy, scipy
  - Machine learning
    - Scikit-learn, tensorfly, keras, pytorch, fastai
  - Geospatial analysis
    - PySAL, GDAL, Shapely, Fiona, …
  - Image processing and computer vision
    - PIL, OpenCV, scikit-image

Why Python?

- Popularity
- Productivity
- Interoperability
- Solves the “two-language” problem
- Scientific Python ecosystem
- Community





Data Science workflows - Deployment

- Deploy as information products
  - WebMaps
  - Web scenes
  - Layers
- Deploy as web tools
  - Geoprocessing script tools
  - Binder projects
- Deploy as dashboards
  - ArcGIS Operations Dashboard
  - Jupyter dashboard
  - Plotly dashboard

Questions?


esri
THE SCIENCE OF WHERE

Print Your Certificate of Attendance

Print Stations Located at L Street Bridge

Tuesday

12:30 pm – 6:30 pm
GIS Solutions Expo
Hall D

5:15 pm – 6:30 pm
GIS Solutions Expo Social
Hall D

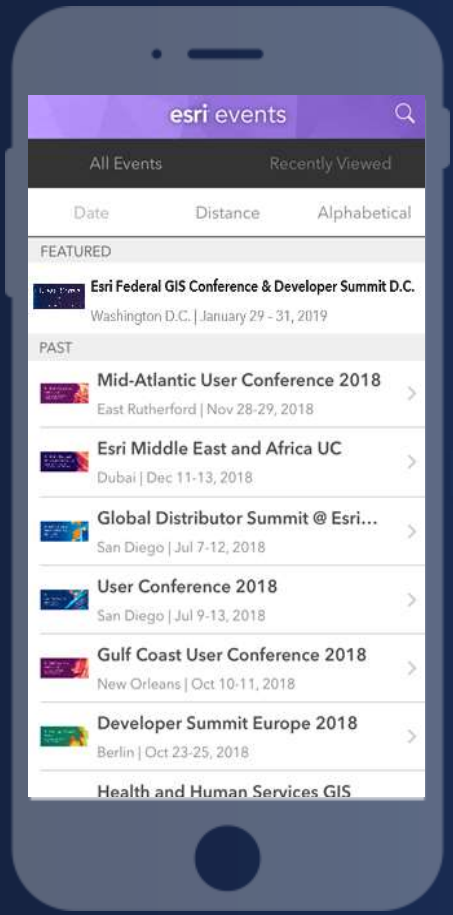
Wednesday

10:45 am – 5:15 pm
GIS Solutions Expo
Hall D

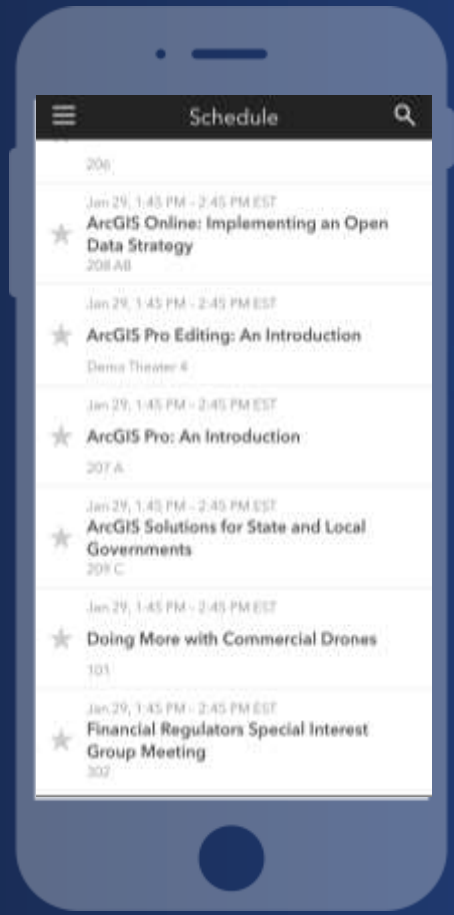
6:30 pm – 9:00 pm
Networking Reception
National Museum of Natural History

Please Take Our Survey on the App

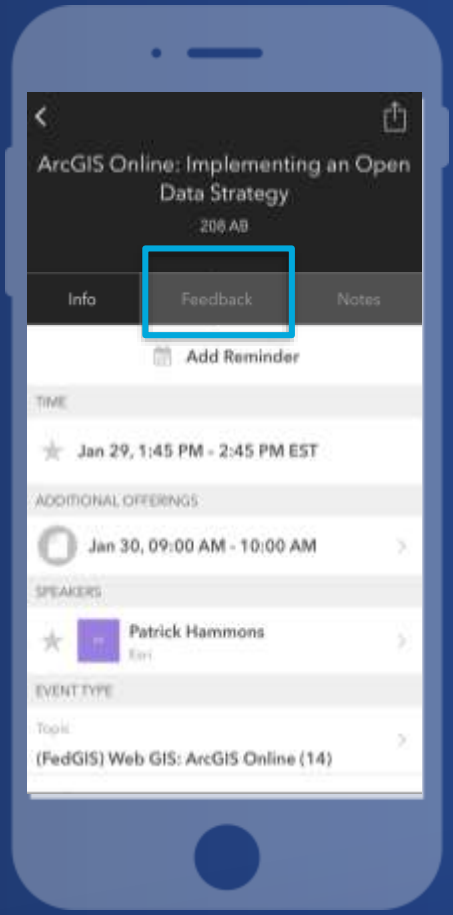
Download the Esri Events app and find your event



Select the session you attended



Scroll down to find the feedback section



Complete answers and select “Submit”