

GeoAI with ArcGIS Pro

Cedric Despierre Corporon | Rami Alouta

GIS for a Sustainable World Conference



GeoAI With ArcGIS Pro

Agenda

- What is GeoAI
- What can it do?
- Where can you apply GeoAI (Sneak peak on the GP tools)
- A deeper dive into GeoAI within ArcGIS
- Getting Started | Environment & Licensing
- GeoAI in ArcGIS Pro
 - GUI Demo
- Advanced GeoAI Workflows with Python
 - Notebook Demo

What is GeoAI?



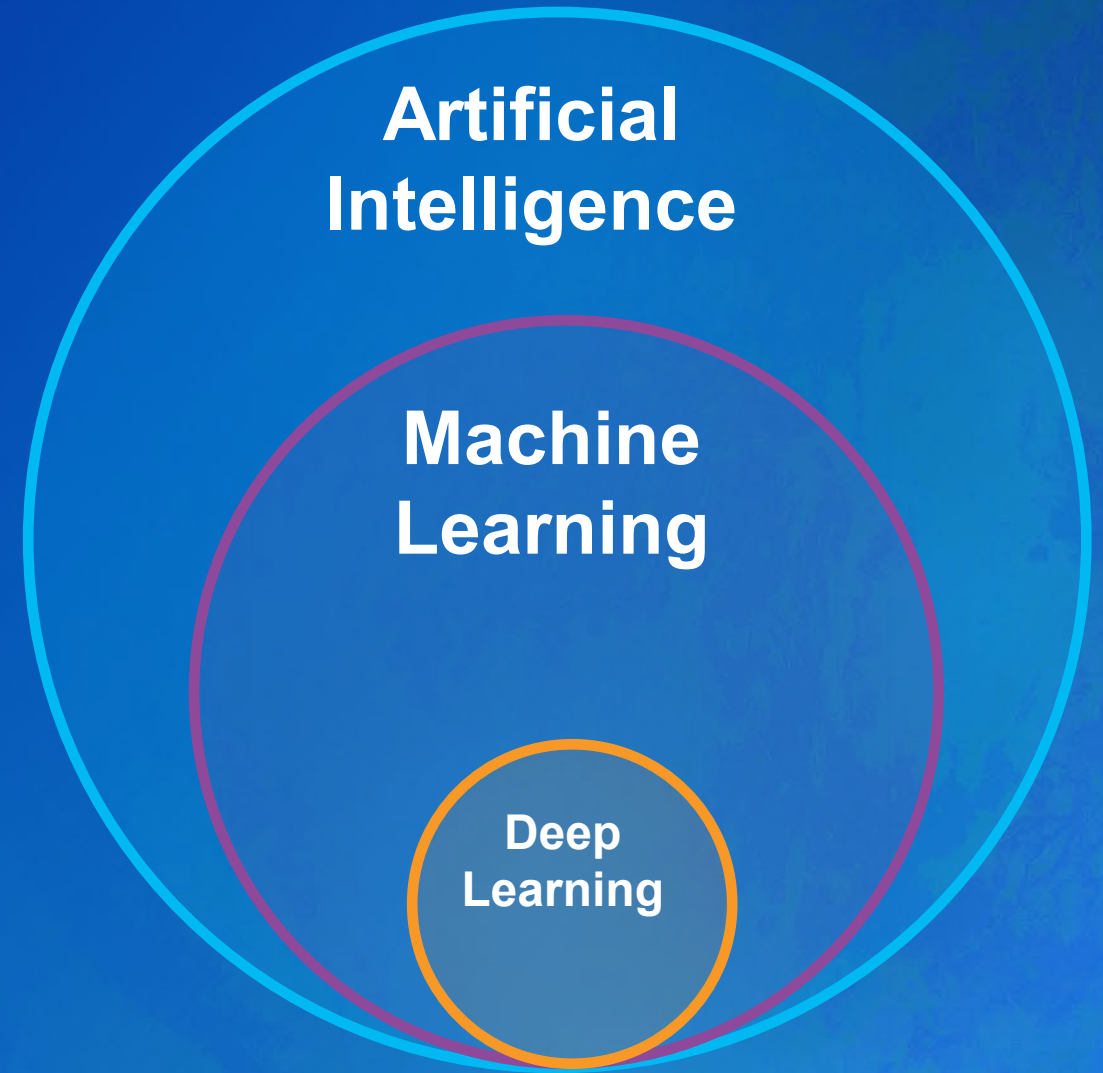
What Is AI?

Summary:

Really, machine learning (ML)

Machine learning is about extracting patterns from data to derive rules, instead of these rules being explicitly programmed.

Deep learning is a type of ML using deep neural networks to find complex patterns especially in unstructured data (such as images, text, voice, and lidar).



GeoAI in ArcGIS



Things to know

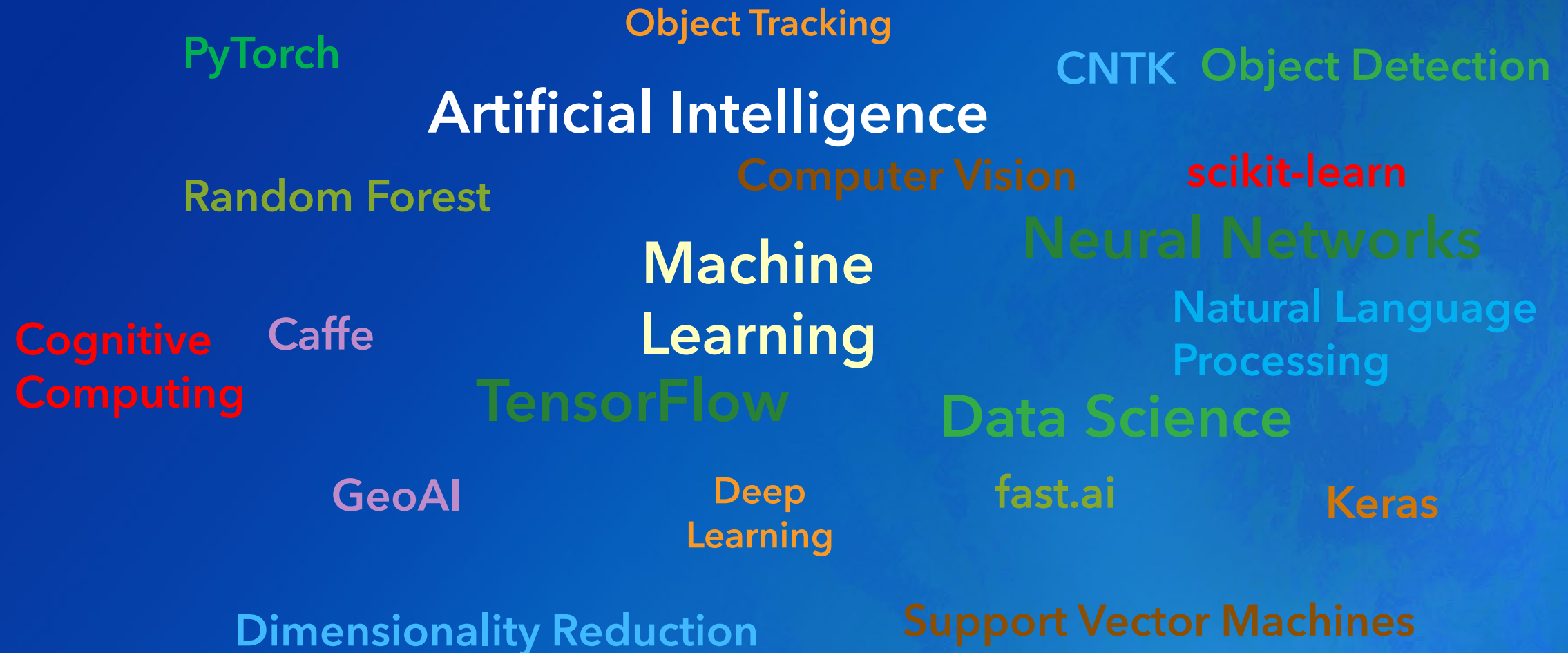
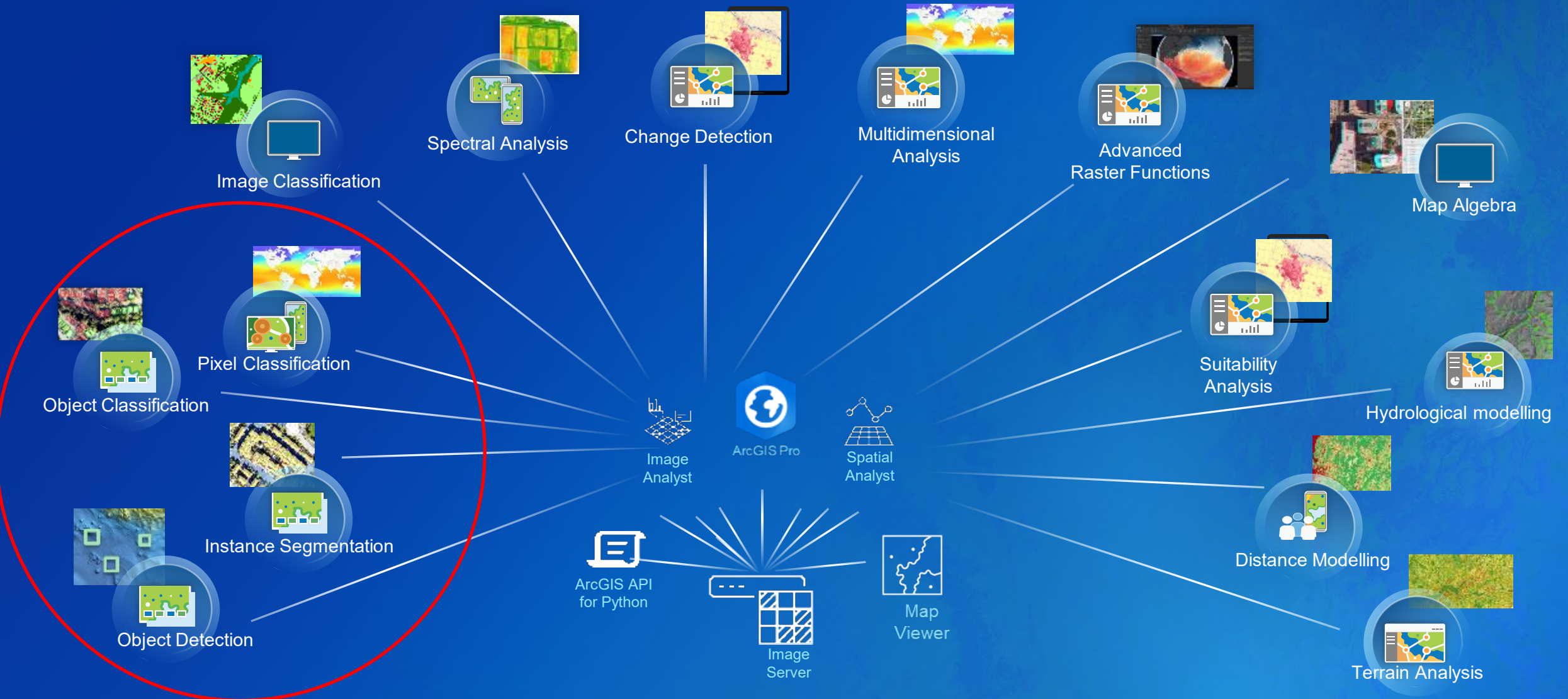


Image Analysis Extracting Information From Imagery



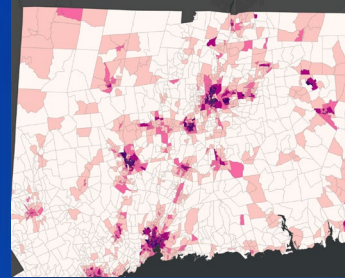
What can GeoAI Do?



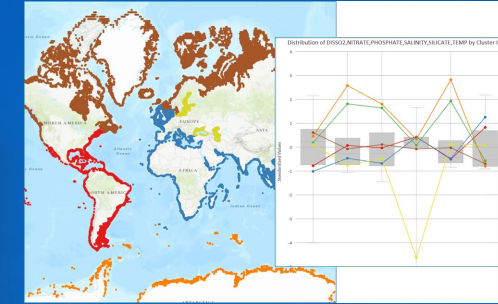
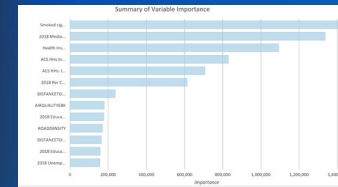
What Can Machine Learning Do?



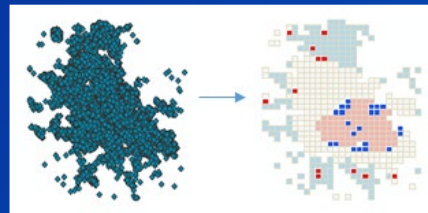
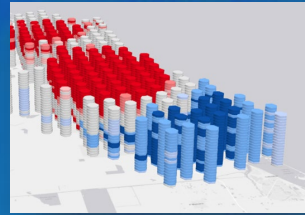
Extract features from
Imagery & LiDAR



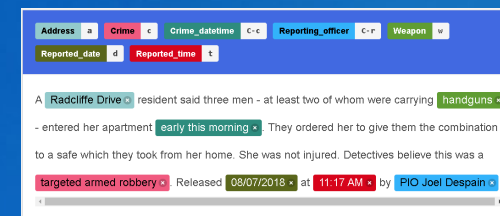
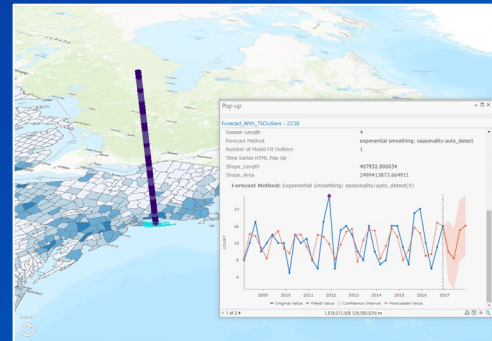
Make predictions



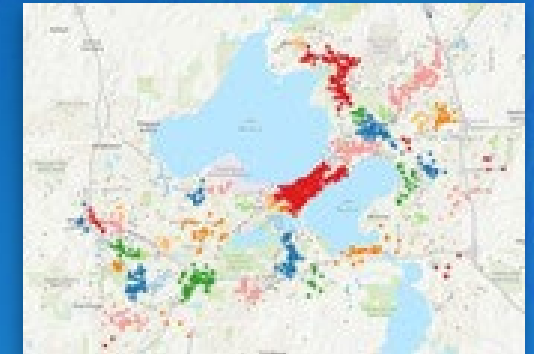
Find patterns & clusters



Detect anomalies



Extract insights from
unstructured text



Object Detection, Pixel Classification, Object Classification from Imagery

Building footprints



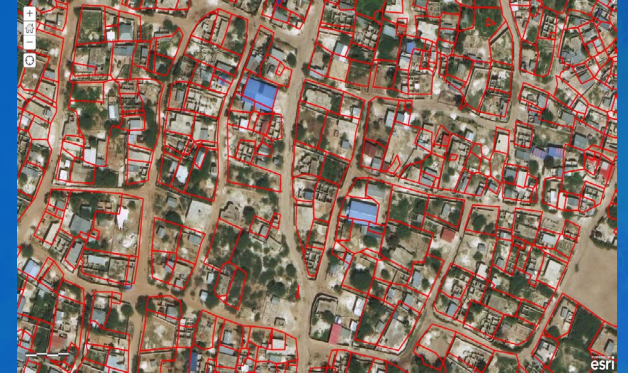
Roads



Land cover



Parcel boundaries



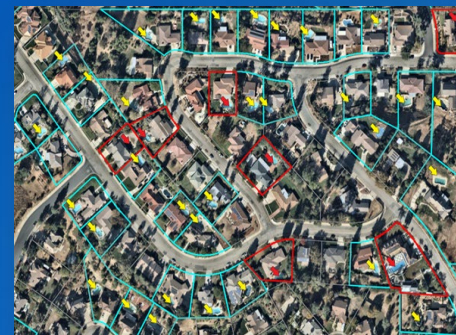
Palm trees



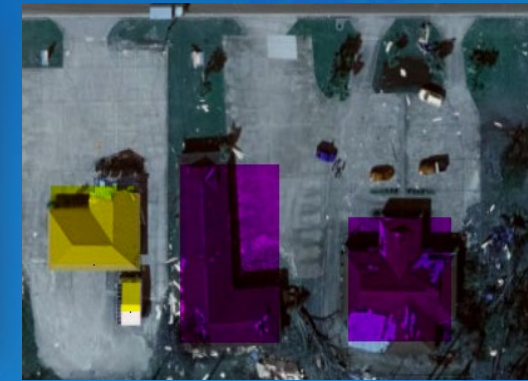
Oil pads



Swimming pools



Damaged structures



DL will detect what you can see in the imagery

Where can you apply GeoAI



AI is not one product. It spans the ArcGIS system.



Where we offer machine learning integration.



ArcGIS API for Python

ArcGIS Velocity

ArcGIS Notebooks

ArcGIS Pro

ArcGIS Online

ArcGIS Enterprise

ArcGIS Hub - [Citizen Data Science](#)

ArcGIS QuickCapture - [Edge AI \(in R&D\)](#)

ArcGIS Insights

ArcGIS Pro for Intelligence

A deeper dive into GeoAI within ArcGIS



Prepare Data, Make Predictions, Find Correlations, Understand Top Variables, and More

A map of the Eastern United States and parts of Canada. The Appalachian Mountains are highlighted in green and run from the northeast down to the Gulf of Mexico. Major cities marked include Toronto, Chicago, Detroit, St. Louis, Dallas, Houston, Atlanta, Philadelphia, Washington, Miami, and Havana. The Gulf of Mexico is labeled at the bottom.

Forecast Method: exponential smoothing; seasonality:user_defined(12)

CONSOMMATION EN MILLE TONNES

60,000,000
55,000,000
50,000,000
45,000,000
40,000,000
35,000,000
30,000,000
25,000,000
20,000,000
15,000,000
10,000,000

2001 2002 2003 2004 2005 2006 2007 2008 2009 2010

— Original Value — Fitted Value — Confidence Interval — Forecasted Value

The screenshot shows the ArcGIS Online interface. The map displays the United States with a green and white pattern, representing SNAP participation. The interface includes a search bar at the top, a layer list on the left, and a table of contents on the right. The map is titled 'SNAP Participation' and shows a green and white pattern across the United States. The table of contents on the right lists various layers, including 'Count of people within the SNAP participation area' and 'Count of people within the SNAP participation area'.

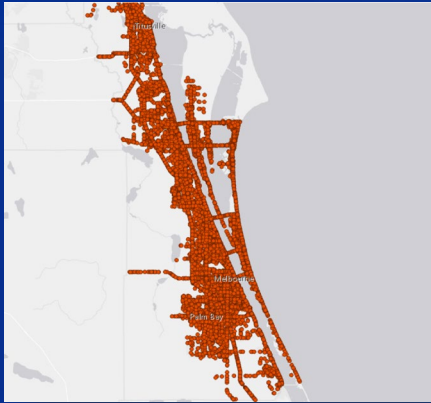
Summary of Variable Importance

Variable	Importance (approx.)
Smoked cig...	1,400,000
2018 Media...	1,350,000
Health Ins...	1,100,000
ACS HHs: In...	850,000
ACS HHs: L...	750,000
2018 Per C...	650,000
DISTANCETO...	250,000
AIRQUALITYEBK	180,000
2018 Educa...	180,000
ROADDENSITY	180,000
DISTANCETO...	150,000
2018 Educa...	150,000
2018 Unemp...	150,000

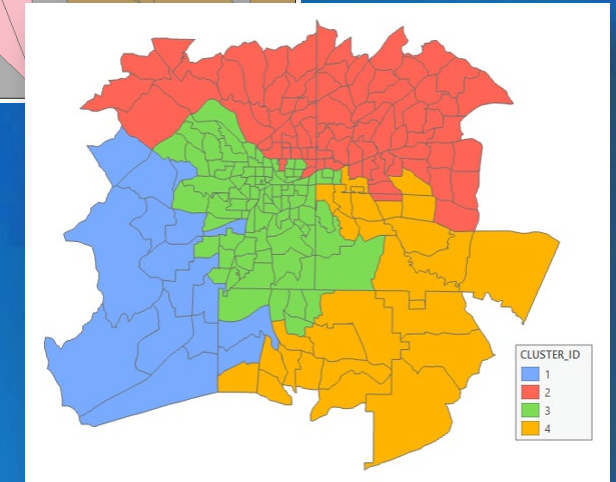
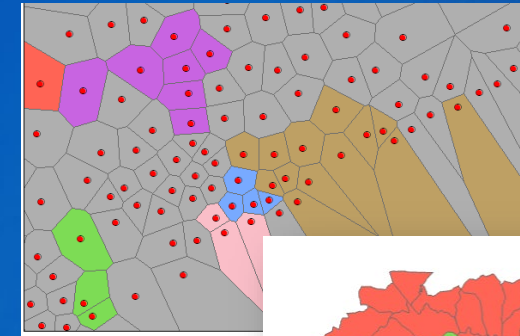
Chart of kc_house_data

2. Pattern Mining and Clustering

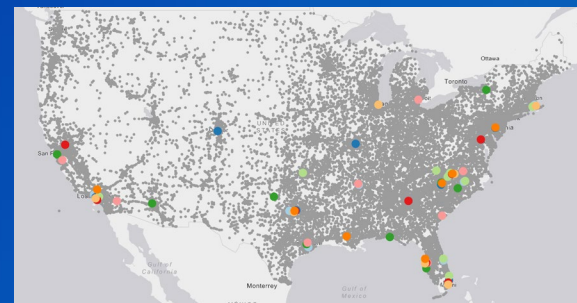
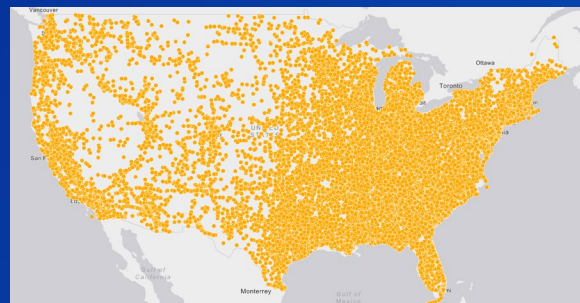
Understand Natural Groupings in Data That Are Statistically Significant



Emerging and fading hotspots for Crashes using **SpaceTime Pattern Mining Toolbox**



Find spatially contiguous clusters for animal territories using **Spatially Constrained Multivariate Clustering**

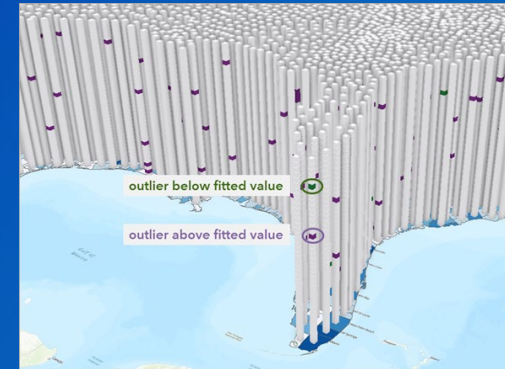


Most important fatal crashes clusters using **Density Based Clustering (DBScan)**

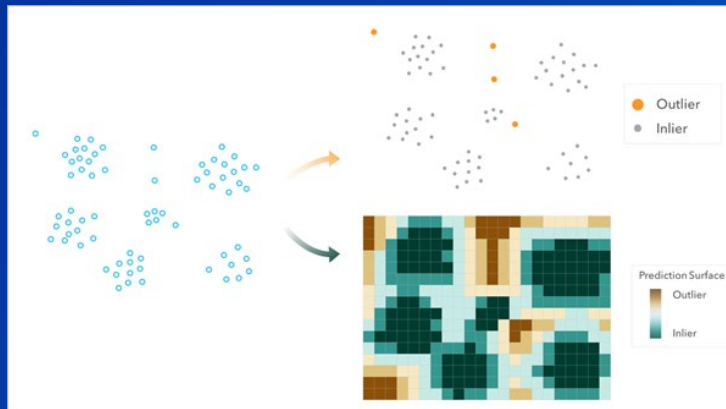
3. Anomaly Detection

Spatial, Temporal, and Spatiotemporal Outliers

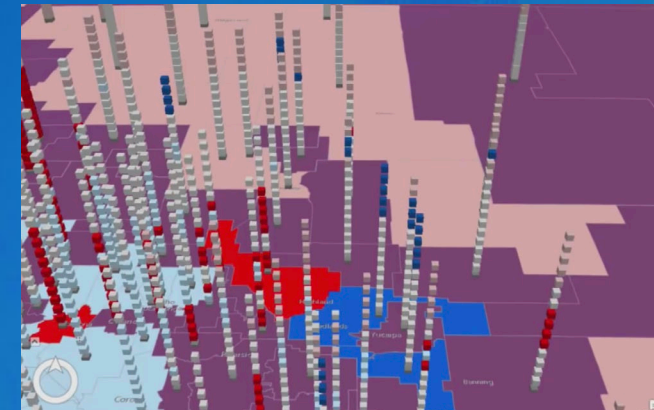
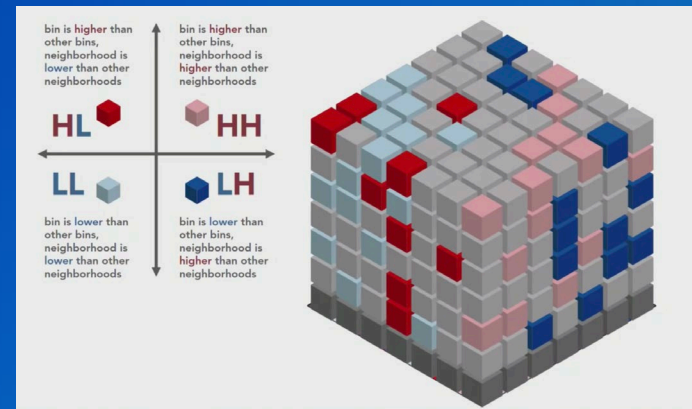
Time series outliers (temporal)



Spatial outliers (spatial)

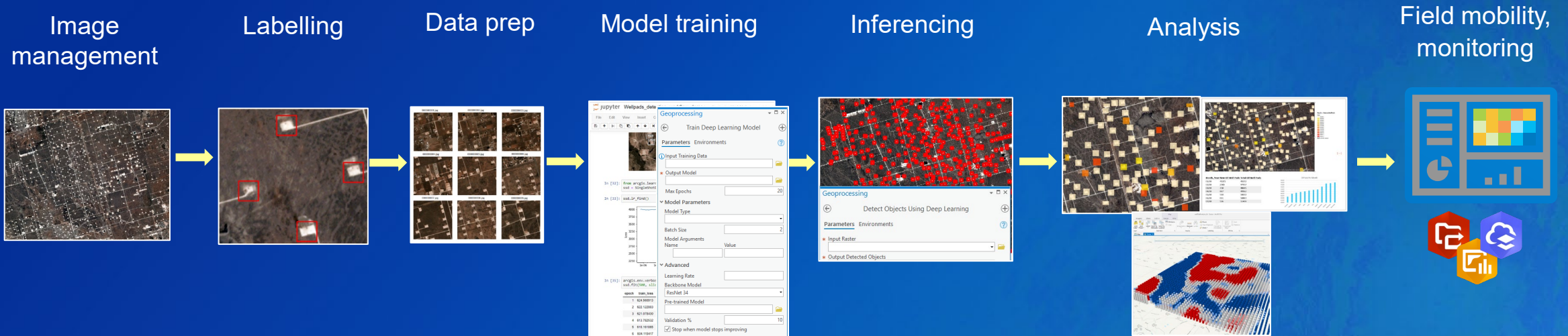


Local outlier analysis (spatiotemporal)



Imagery AI: End-to-End Workflow

Extract Insights from Imagery at Scale, with High Speed and Accuracy



For Wide Range of Data Types


- Aerial
- Satellite
- Radar
- Lidar
- Motion imagery
- Bathymetry
- Point cloud
- Drone

Implementing Many Tasks

- Object classification
- Object detection
- Pixel classification
- Image translation
- Object tracking
- Scanned maps

Pre-trained Models on ArcGIS Living Atlas

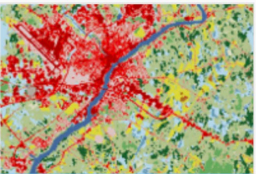
Plug-and-Play Models. No Training Needed. Easy Re-training Using Local Data.



Deep Learning Package

Building Footprint Extraction - USA


By esri_analytics



Deep Learning Package

Land Cover Classification (Land Use)


By esri_analytics



Car Detection - USA

Deep Learning Package By esri_analytics

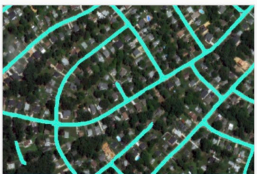
Deep learning model to detect cars in high resolution imagery.



Solar Panel Detection - USA

Deep Learning Package By esri_analytics


Deep learning model to detect solar panels in high resolution imagery.



Deep Learning Package

Road Extraction - North America


By esri_analytics



Deep Learning Package

Human Settlements Classification


By esri_analytics



Deep Learning Package

Solar Panel Detection - USA


By esri_analytics



Deep Learning Package

Parcel Extraction - USA


By esri_analytics



Deep Learning Package

Land Cover Classification (Seasonal)


By esri_analytics



Pool Detection - USA

Deep Learning Package By esri_analytics

Deep learning model to detect swimming pools in high-resolution aerial or satellite imagery.



Object Tracking

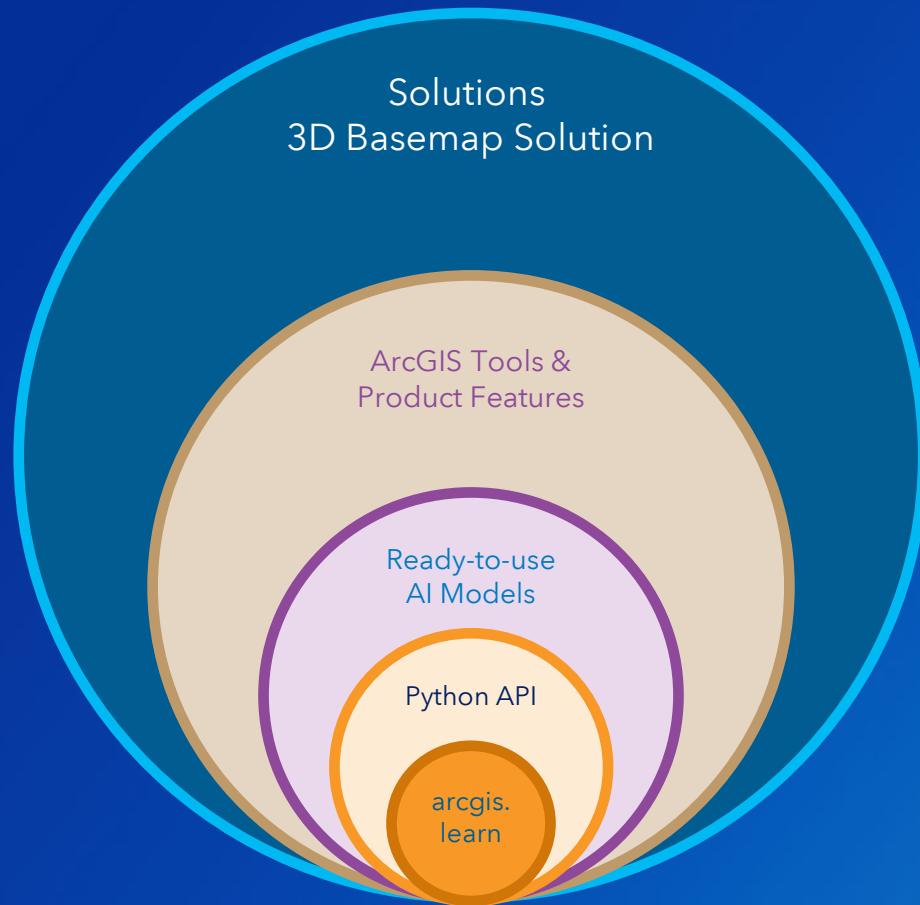
Deep Learning Package By esri_analytics

Deep learning model for tracking objects in motion imagery. This model was trained using DAVIS dataset and further fine-tuned on aerial motion imagery.

Use Models Within

- ArcGIS Pro (+ Image Analyst Extension)
- ArcGIS Enterprise (+ Image Server)
- ArcGIS Online (+ ArcGIS Image)

3D Basemap Solution – LOD2 Building Extraction

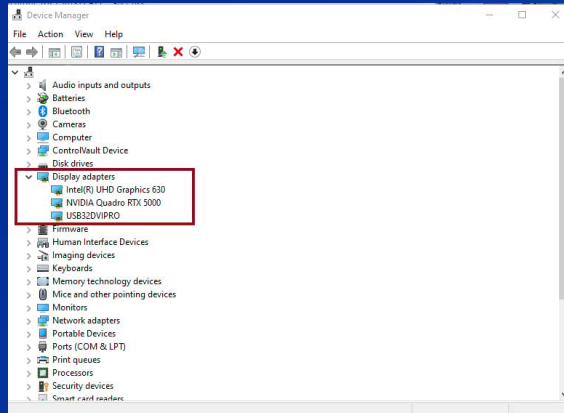


Getting Started | Environment & Licensing



Getting Started

Prerequisites



GPU availability, specifications,
and compatibility

```
C:\Windows\system32\cmd.exe
Fri Feb 19 16:26:50 2021

+-----+
| NVIDIA-SMI 451.82      Driver Version: 451.82      CUDA Version: 11.0      |
+-----+-----+
| GPU   Name                TCC/WDDM | Bus-Id      Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap |      Memory-Usage | GPU-Util  Compute M. |
+-----+-----+
| 0   Quadro RTX 5000      WDDM    | 00000000:01:00:0 | Off      N/A      |
| N/A   51C    P8      27W /  N/A    | 214MiB / 16384MiB |    0%      Default  |
+-----+-----+

Processes:
GPU   GI   CI          PID  Type   Process name                      GPU Memory
  ID   ID                                     Usage
+-----+-----+
No running processes found
+-----+-----+
```

CUDA | nvidia-smi

Licensing

ArcGIS Pro Single Use License

Name	Licensed	Expires
Basic	No	N/A
Standard	No	N/A
Advanced	Yes	10/19/2022

Esri Extensions

Name	Licensed	Expires
Business Analyst	Yes	10/19/2022
Data Interoperability	Yes	10/19/2022
Data Reviewer	Yes	10/19/2022
Defense Mapping	Yes	10/19/2022
Geostatistical Analyst	Yes	10/19/2022
Image Analyst	Yes	10/19/2022
Indoors	Yes	10/16/2022
LocateXT	Yes	10/19/2022
Location Referencing	Yes	10/19/2022

ArcGIS Pro availability and version



Deep learning framework for
ArcGIS Pro installation

```
Python

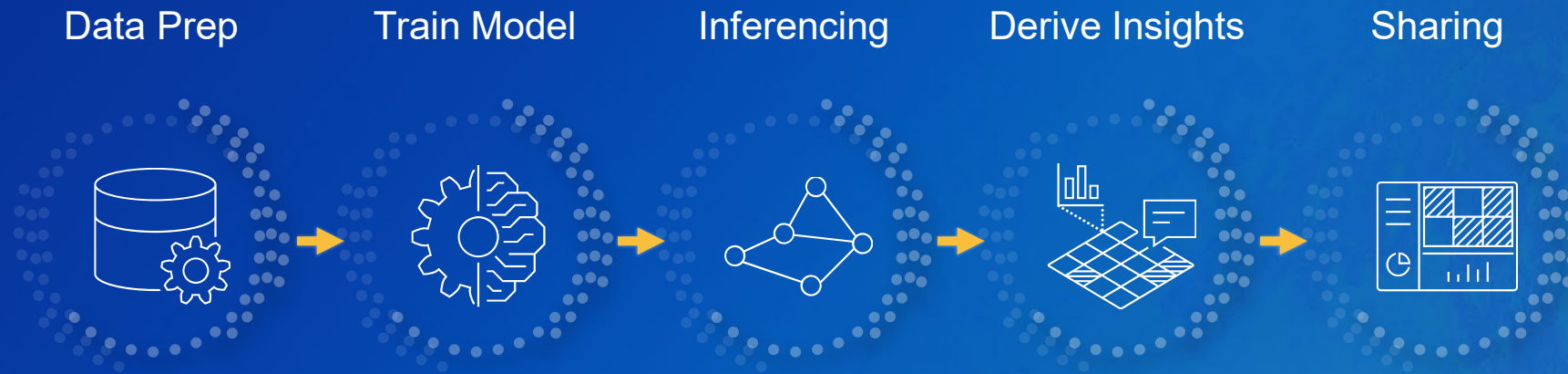
import torch
import fastai
torch.cuda.is_available()
True
```

Communicating with the GPU

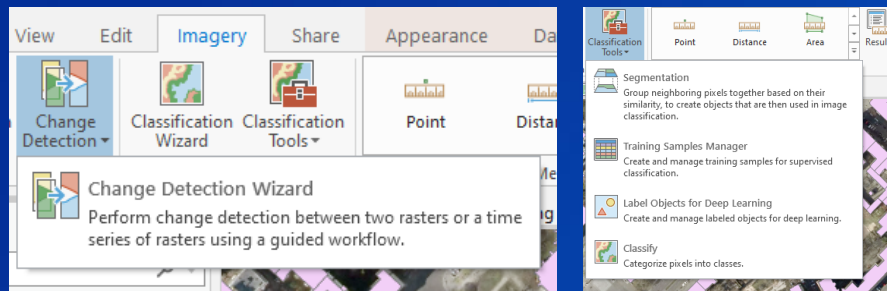
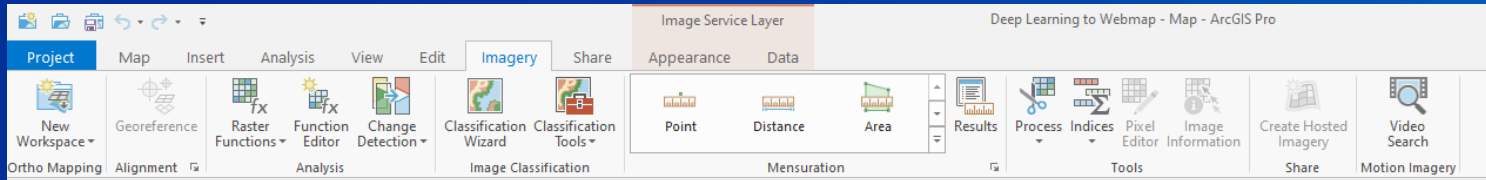
GeoAI in ArcGIS Pro



Machine Learning Lifecycle



GeoAI Tools in ArcGIS Pro



Deep Learning

- ✎ Classify Objects Using Deep Learning
- ✎ Classify Pixels Using Deep Learning
- ✎ Compute Accuracy For Object Detection
- ✎ Detect Change Using Deep Learning
- ✎ Detect Objects Using Deep Learning
- ✎ Export Training Data For Deep Learning
- ✎ Non Maximum Suppression
- 📄 Train Deep Learning Model

Image Analyst Tools

- ▶ ✎ Change Detection
- ▶ ✎ Classification and Pattern Recognition
- ▶ ✎ Deep Learning
- ▶ ✎ Extraction
- ▶ ✎ Map Algebra
- ▶ ✎ Math
- ▶ ✎ Motion Imagery
- ▶ ✎ Multidimensional Analysis
- ▶ ✎ Overlay
- ▶ ✎ Statistical

Change Detection

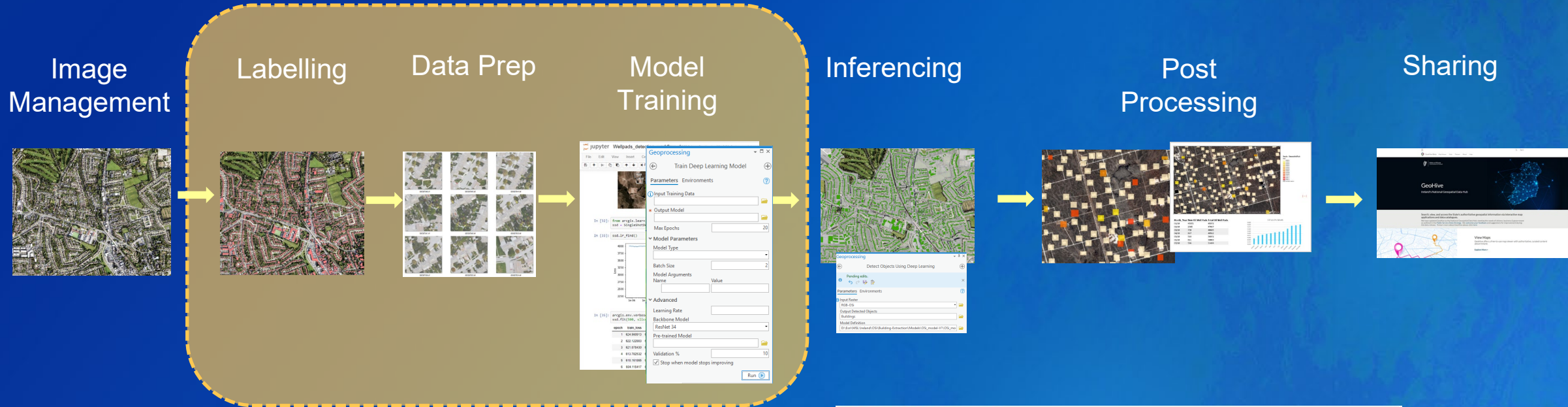
- ✎ Analyze Changes Using CCDC
- ✎ Analyze Changes Using LandTrendr
- ✎ Compute Change Raster
- ✎ Detect Change Using Change Analysis Raster

Classification and Pattern Recognition

- ✎ Classify Raster
- ✎ Compute Confusion Matrix
- ✎ Compute Segment Attributes
- ✎ Create Accuracy Assessment Points
- ✎ Generate Training Samples From Seed Points
- ✎ Inspect Training Samples
- ✎ Linear Spectral Unmixing
- ✎ Predict Using Regression Model
- ✎ Remove Raster Segment Tiling Artifacts
- ✎ Segment Mean Shift
- ✎ Train ISO Cluster Classifier
- ✎ Train K-Nearest Neighbor Classifier
- ✎ Train Maximum Likelihood Classifier
- ✎ Train Random Trees Classifier
- ✎ Train Random Trees Regression Model
- ✎ Train Support Vector Machine Classifier
- ✎ Update Accuracy Assessment Points

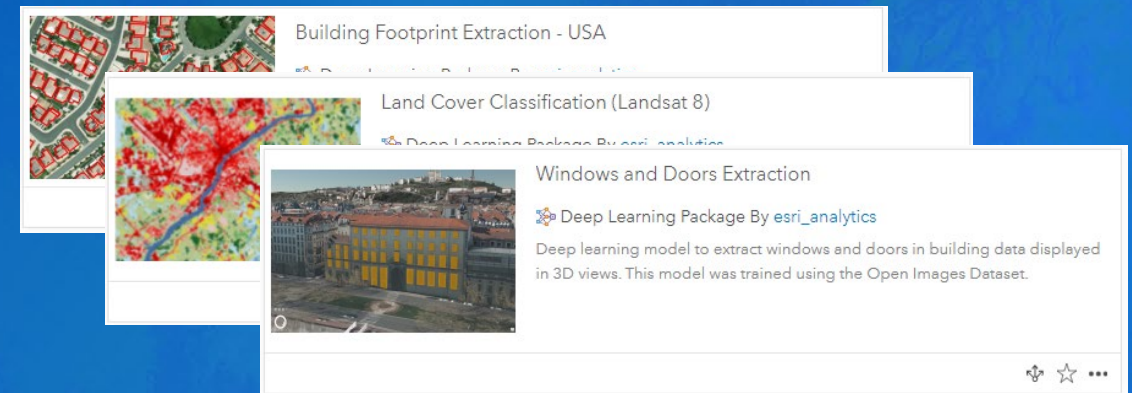
Deep Learning Workflow

Pre-trained Models



Eliminates:

- Imagery requirements for model training
- Labelling requirements
- Training AI models
- Massive compute requirements

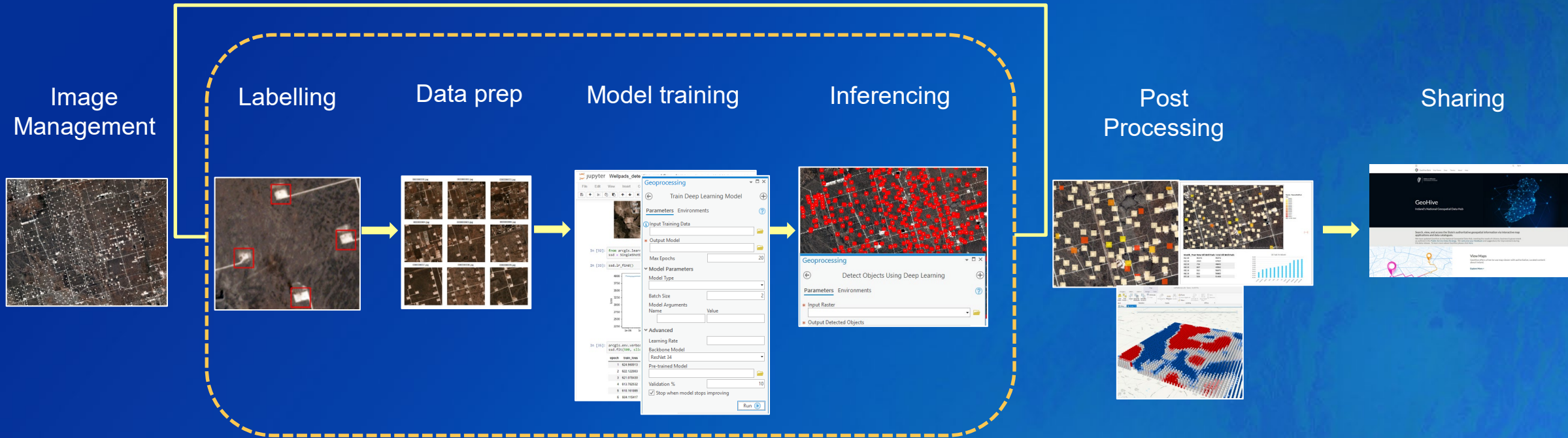




Applying DL with Out of the Box Model

Building Detection

Fine-Tuning Mask RCNN Building Detection Model.



For Wide Range of Data Types

- Aerial
- Satellite
- Radar
- Lidar
- Motion imagery
- Bathymetry
- Point cloud
- Drone

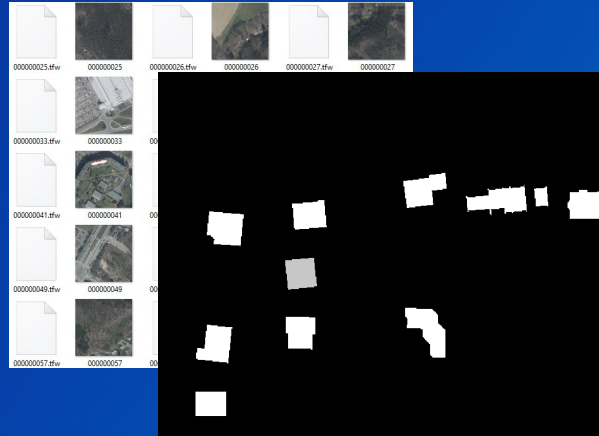
Implementing Many Tasks

- Object classification
- Object detection
- Pixel classification
- Image translation
- Object tracking
- Scanned maps

Fine-Tuning Mask RCNN Building Detection Model.



Label with existing buildings



Data Prep

```
In [9]: with arcpy.EnvManager(scratchworkspace=temp_gdb, processorType="GPU", workspace_temp_gdb):
        TrainDeepLearningModel(training_data=model_output, 100, "MASKRCNN", 4, "chip_size 512",
                                None, "RESNET50", pretrained_dlpk, 10, "STOP_TRAINING", "FREEZE_MODEL")

epoch      train_loss  valid_loss  time
0          1.394653  1.004157  11:58
1          1.099954  1.051891  12:03
2          1.020073  1.040416  12:02
3          1.001396  1.022784  12:03
4          0.975865  1.013227  11:17
5          0.976907  1.005356  11:12
6          0.987169  0.993153  11:08
7          0.968420  0.990451  11:08
8          0.985940  0.977256  11:11
9          0.943040  0.967820  11:08
10         0.933337  0.966996  11:09
11         0.931865  0.966357  11:09
12         0.884022  0.966551  11:10
13         0.899677  0.951055  11:08
14         0.920003  0.953767  11:09
15         0.864720  0.948522  11:14
16         0.915749  0.940785  11:11
17         0.846029  0.940156  11:09
18         0.877141  0.931010  11:10
19         0.853407  0.941983  11:09
20         0.874127  0.930044  11:10
21         0.875703  0.923362  11:10
22         0.880115  0.927745  11:11
23         0.833817  0.921910  11:10
24         0.953438  0.920121  11:09
25         0.864735  0.924087  11:12
26         0.888156  0.924411  11:09
27         0.876759  0.923098  11:10
28         0.838580  0.938189  11:16
29         0.827601  0.929988  11:09
Epoch 29: early stopping
Computing model metrics...
```

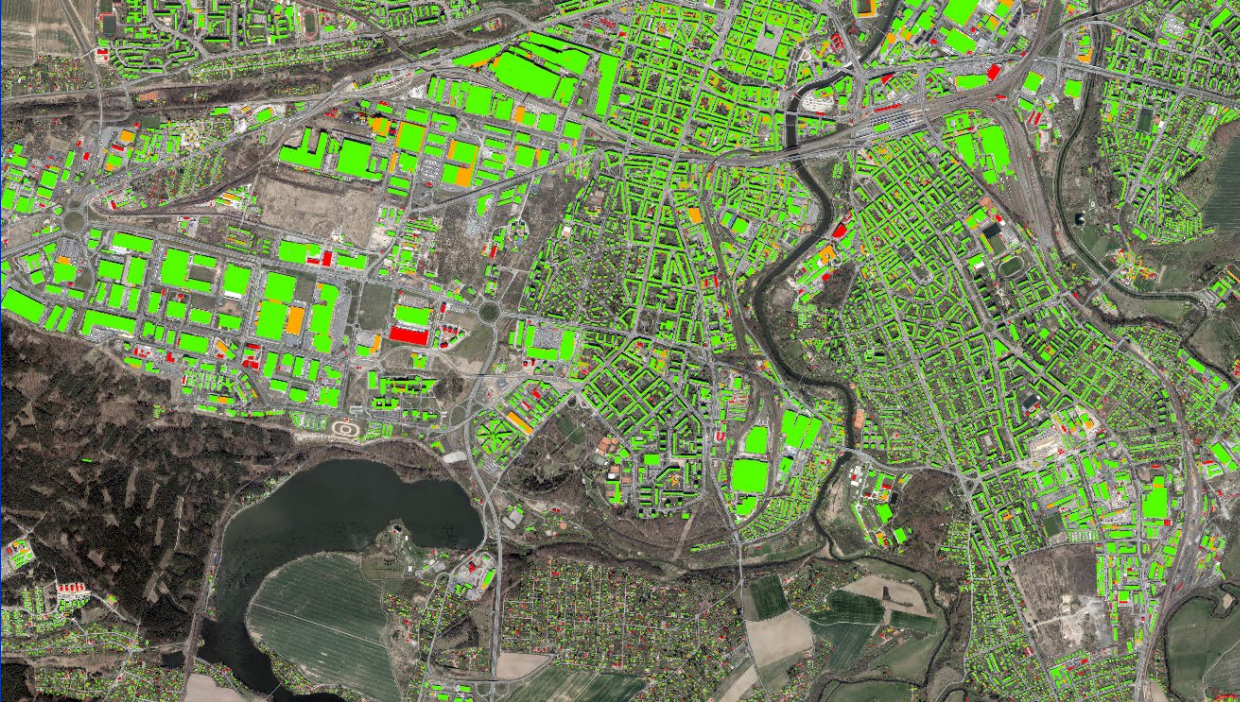
Model training (fine-tuning)



Inference with fine-tuned model (71500 features)

Fine-Tuning the Deep Learning Package

Building Detection



Advanced GeoAI Workflows with Python



Automation end to end workflow with ArcGIS Pro Notebooks.

- Utilize ArcPy & Python API to automate process

```
deep_learning_workflow X
Edit View Insert Cell Help ArcGISPro

In [7]: with arcpy.EnvManager(cellSize=0.3):
        ExportTrainingDataForDeepLearning("temp_mosaic", training_data, training_dataset, 'TIFF',
        '512', '512', '0', '0', "ONLY_TILES_WITH_FEATURES",
        'RCNN_Masks', 0, 'class', 0, 'None', 0, "MAP_SPACE",
        "PROCESS_AS_MOSAICKED_IMAGE", "NO_BLACKEN", "FIXED_SIZE")

In [9]: with arcpy.EnvManager(scratchWorkspace=temp_gdb, processorType="GPU", workspace=temp_gdb):
        TrainDeepLearningModel(training_data,model_output, 100, "MASKRCNN", 4, "chip_size 512",
        None, "RESNET50", pretrained_dlpk, 10, "STOP_TRAINING", "FREEZE_MODEL")

epoch   train_loss  valid_loss  time
0       1.194653   1.084157   11:58
1       1.099954   1.051891   12:03
2       1.028073   1.040416   12:02
3       1.061396   1.022784   12:03
4       0.975865   1.013227   11:17
5       0.976907   1.005356   11:12
6       0.987169   0.993153   11:08
7       0.960429   0.990451   11:08
8       0.985940   0.977256   11:11
9       0.943040   0.967826   11:08
10      0.935337   0.966996   11:09
11      0.931865   0.966357   11:09
12      0.884022   0.966551   11:10
13      0.899677   0.951055   11:08
14      0.920003   0.963767   11:09
15      0.864720   0.948522   11:14
16      0.915749   0.949785   11:11
17      0.846629   0.940156   11:09
18      0.877141   0.931010   11:10
19      0.863407   0.941983   11:09
20      0.874127   0.930844   11:10
21      0.875783   0.923362   11:10
22      0.880115   0.927745   11:11
23      0.833817   0.921910   11:10
24      0.953438   0.929121   11:09
25      0.844735   0.924087   11:12
26      0.888156   0.924411   11:09
27      0.876759   0.923898   11:10
28      0.838589   0.930189   11:16
29      0.827601   0.929988   11:09
Epoch 29: early stopping
Computing model metrics...
```

```
In [2]: images_directory = r"C:\Users\ramialouta\Documents\ArcGIS\Projects\deeplearning_czech\True_Ortho"
pretrained_dlpk = r"C:\Users\ramialouta\Documents\ArcGIS\Packages\usa_building_footprints.dlpk"
model_parameters = [{"padding 64;batch_size 16;threshold 0.5;return_bboxes False;tile_size 512",
                    "padding 64;batch_size 4;threshold 0.5;return_bboxes False;tile_size 1024",
                    "padding 128;batch_size 16;threshold 0.5;return_bboxes False;tile_size 512",
                    "padding 128;batch_size 4;threshold 0.5;return_bboxes False;tile_size 1024"}]
training_data = r"C:\Users\ramialouta\Documents\ArcGIS\Projects\deeplearning_czech\training\buildings_training_data_v2"
footprints_fc = r"C:\Users\ramialouta\Documents\ArcGIS\Projects\deeplearning_czech\Footprints\SURE_Pilsen_building_Footprints.gdb\Pilsen_buil
model_output = r"C:\Users\ramialouta\Documents\ArcGIS\Projects\deeplearning_czech\models\retrained_buildings_v2"

In [3]: gdb_dir = images_directory.split("\\")[-1]
gdb = f"{gdb_dir[-1]}_temp.gdb"
final_gdb = f"{gdb_dir[-1]}.gdb"
gdb_dir[0] = f"{gdb_dir[0]}\\\"
gdb_dir = functools.reduce(os.path.join,gdb_dir)

training_dataset = "training_dataset"

# Execute CreateFileGDB
temp_gdb = os.path.join(gdb_dir, gdb)
final_gdb = os.path.join(gdb_dir, "deeplearning_czech.gdb")

if arcpy.Exists(temp_gdb):
    featureclasses = arcpy.ListFeatureClasses()
    fsc=[]
    for fc in featureclasses:
        fc=fc.replace(".", "_")
        fsc.append(fc)
else:
    arcpy.CreateFileGDB_management(gdb_dir, gdb)
    arcpy.env.workspace = temp_gdb
    arcpy.env.overwriteOutput = True

output_detected_buildings_fc = os.path.join(temp_gdb, "detected_buildings")
raw_buildings = os.path.join(final_gdb, "raw_detected_buildings")

arcpy.env.overwriteOutput = True
```

```
In [4]: dlpk_name_extesion = model_output.split("\\")[-1]
dlpk = os.path.join(model_output, f"{dlpk_name_extesion}.dlpk")
start_datetime = datetime.now()
detected_outputs = []
for parameter in model_parameters:
    iter_startdt = datetime.now()
    padding = parameter.split(";")[0].split(" ")[-1]
    btachsize = parameter.split(";")[1].split(" ")[-1]
    threshold = str(int(float(parameter.split(";")[2].split(" ")[-1])*100))
    tilesize = parameter.split(";")[3].split(" ")[-1]
    output_fc = f"czech_buildings_30cm_P(padding){padding}.B(btachsize)_Threshold(threshold)_Tile(tilesize)_{dlpk_name_extesion}"
    with arcpy.EnvManager(processorType="GPU", cellSize=0.3, workspace=final_gdb):
        arcpy.la.DetectObjectsUsingDeepLearning("temp_mosaic", output_fc, dlpk,
        parameter, "NO_MMS", "Confidence",
        "Class", 0, "PROCESS_ITEMS_SEPARATELY")

    torch.cuda.empty_cache()
    detected_outputs.append(output_fc)
    iter_enddt = datetime.now()
    iter_minutes_diff = (iter_enddt - iter_startdt).total_seconds() / 60.0
    print(f"The processing of image: {output_fc} took {int(iter_minutes_diff)} minutes")
```

Deep Learning with Python

▼ **Working Environment:**

In [4]: `dl_workflow_env = dlw.workflow_env()`
`dl_workflow_env`

Where to run deep learning workflows:

In [5]: `widgets_box = dlw.data_inputs()`
`widgets_box[0]`

Choose from the below workflows:

- ☒ Fine-tune Living Atlas Model
- ☐ Fine-tune Local Model
- ☐ Train new model
- ☐ Inference with local model
- ☐ Inference with Living Atlas Model

Imagery layer in ArcGIS Pro TOC:

Labelled data in ArcGIS Pro TOC:

ArcGIS Pro project directory:

Deep Learning Resources

Deep Learning with ArcGIS Pro Tips & Tricks:
Part 1



Scan Me

Deep Learning with ArcGIS Pro Tips & Tricks:
Part 2



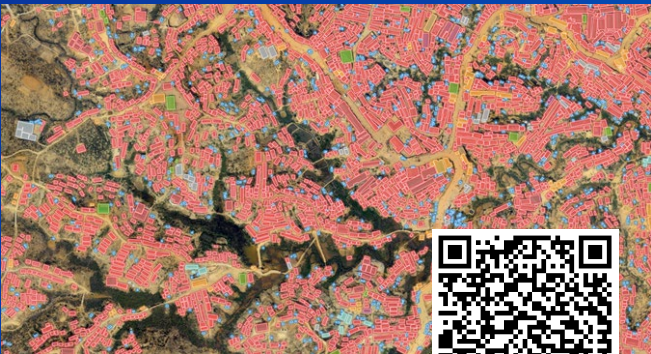
Scan Me

Deep Learning with ArcGIS Pro Part 3: QA/QC
Extracted Features



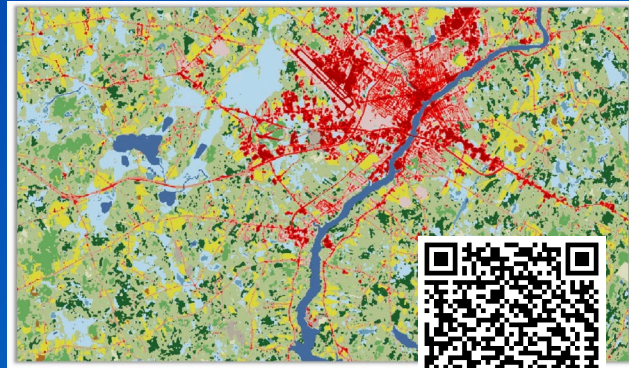
Scan Me

Deep Learning with ArcGIS Pro Part 3: QA/QC
Extracted Features



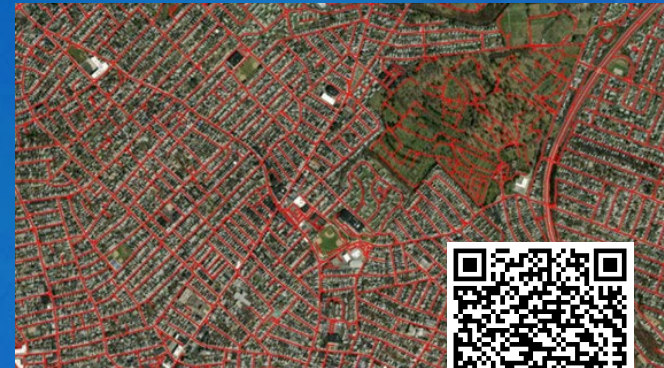
Scan Me

Introducing pre-trained geospatial deep
learning models



Scan Me

Pre-trained deep learning models
update (February 2021)



Scan Me

Thank you to our sponsors

Co-host



United Nations Institute for Training and Research



unitar



UNOSAT



esri®

THE
SCIENCE
OF
WHERE®

