



March 2020

# Arc Hydro: Wetland Identification Model

380 New York Street  
Redlands, California 92373-8100 USA  
909 793 2853  
[info@esri.com](mailto:info@esri.com)  
[esri.com](http://esri.com)



**esri**

THE  
SCIENCE  
OF  
WHERE™

# Table of Contents

Section Title	Page
<b>1.0 Executive summary .....</b>	<b>1</b>
1.1 Document history .....	1
<b>2.0 Getting started with WIM .....</b>	<b>2</b>
<b>3.0 Solution overview .....</b>	<b>4</b>
3.1 Database design .....	6
3.2 Output data naming conventions .....	7
3.3 Tool operations .....	9
3.3.1 Preprocessing: <i>Smooth High-Resolution DEM</i> .....	9
3.3.2 Preprocessing: <i>Hydrocondition High-Resolution DEM</i> .....	12
3.3.3 Predictor Variable Calculation: <i>Calculate Depth to Water Index</i> .....	13
3.3.4 Predictor Variable Calculation: <i>Calculate Curvature</i> .....	16
3.3.5 Predictor Variable Calculation: <i>Calculate Topographic Wetness Index</i> .....	18
3.3.6 Classification and Accuracy Assessment: <i>Train Test Split</i> .....	21
3.3.7 Classification and Accuracy Assessment: <i>Train Random Trees</i> .....	24
3.3.8 Classification and Accuracy Assessment: <i>Run Random Trees</i> .....	28
3.3.9 Classification and Accuracy Assessment: <i>Assess Accuracy</i> .....	33
3.3.10 WIM tools as an ArcGIS Pro model .....	39
3.4 Results interpretation .....	47
<b>4.0 Individual tool help .....</b>	<b>49</b>
4.1 Smooth High-Resolution DEM .....	49
4.1.1 Input High-Resolution DEM .....	49
4.1.2 Smoothing Method .....	49
4.1.3 Output Smoothed DEM .....	49
4.1.4 Rate of Smoothing: Smoothing Width or Perona-Malik Iterations (Optional) .....	49
4.2 Hydrocondition High-Resolution DEM .....	51
4.2.1 Input High-Resolution DEM .....	51
4.2.2 Conditioning Method .....	51
4.2.3 Output Conditioned DEM .....	51
4.3 Calculate Depth to Water Index .....	52
4.3.1 Input Smoothed DEM .....	52

4.3.2	Output DTW Raster.....	52
4.3.3	Save Intermediate Outputs .....	52
4.3.4	Output DTW Slope Raster .....	52
4.3.5	Input DTW Slope Raster (Optional) .....	52
4.3.6	Input Surface Water Raster (Optional).....	53
4.4	Calculate Curvature .....	54
4.4.1	Input Smoothed DEM.....	54
4.4.2	Output Curvature Raster .....	54
4.5	Calculate Topographic Wetness Index .....	55
4.5.1	Input Hydroconditioned DEM .....	55
4.5.2	Output TWI Raster .....	55
4.5.3	Save Intermediate Outputs .....	55
4.5.4	Output TWI Slope.....	55
4.5.5	Output TWI Specific Catchment Area .....	55
4.5.6	Input TWI Slope Raster (Optional) .....	56
4.5.7	Input TWI Specific Catchment Area Raster (Optional) .....	56
4.6	Train Test Split.....	57
4.6.1	Input Ground Truth Data .....	57
4.6.2	Class Values .....	57
4.6.3	Percent to Sample from Each Class .....	57
4.6.4	Output Training Raster.....	57
4.6.5	Output Testing Raster .....	57
4.6.6	Training Sampling Area Constraint (Optional) .....	57
4.7	Train Random Trees .....	58
4.7.1	Input Training Raster.....	58
4.7.2	Input Predictor Variables Raster .....	58
4.7.3	Output Trained Model .....	58
4.7.4	Prepared Predictor Variable Raster .....	58
4.7.5	Output Variable Importance .....	58
4.7.6	Number of Trees (Optional) .....	59
4.7.7	Maximum Tree Depth (Optional).....	59
4.7.8	Maximum Number of Features (Optional).....	59
4.7.9	Class Weights (Optional) .....	59
4.8	Run Random Trees.....	60
4.8.1	Input Predictor Variables Raster .....	60

4.8.2	Input Trained Model .....	60
4.8.3	Output Prediction Raster .....	60
4.8.4	Save Probability Rasters .....	60
4.9	Assess Accuracy.....	61
4.9.1	Input Testing Raster .....	61
4.9.2	Input Prediction Raster .....	61
4.9.3	Output Metrics Directory .....	61
4.9.4	Optional Accuracy Outputs (Optional).....	61
<b>5.0</b>	<b>References .....</b>	<b>62</b>

## 1.0 Executive summary

The Wetland Identification Model (WIM) is an automated geoprocessing workflow created through original research at the University of Virginia. The workflow derives topographic indices from a high-resolution DEM and uses these as inputs to a machine learning algorithm to predict the areal extent of wetlands. The creation and testing of the WIM resulted in 2 peer-reviewed journal articles (O’Neil et al., 2018; O’Neil et al., 2019). Although originally created as a python workflow using various open source libraries, the WIM is now available as an ArcGIS toolset operating on the Esri software, specifically ArcGIS Pro 2.5 and higher. This document describes the implementation of WIM within the Arc Hydro Pro framework.

## 1.1 Document history

**Table 1. Document Revision History**

Version	Description	Date
1	Initial document (GLO).	March 2020

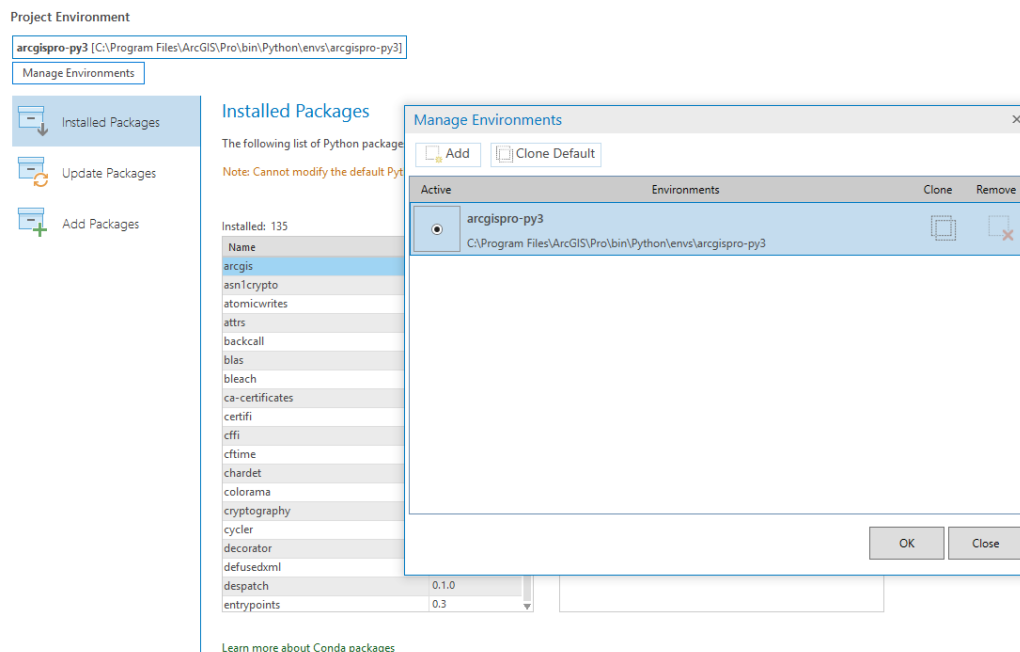
## 2.0 Getting started with WIM

The WIM is included in the installation of Arc Hydro Pro 2.5. Guidelines for installing Arc Hydro can be found [here](#). Users should install version 2.0.165 or higher of Arc Hydro from <http://downloads.esri.com/archydro/ArcHydro/Setup/Pro/>. Testing data used in this documentation can be found [here](#).

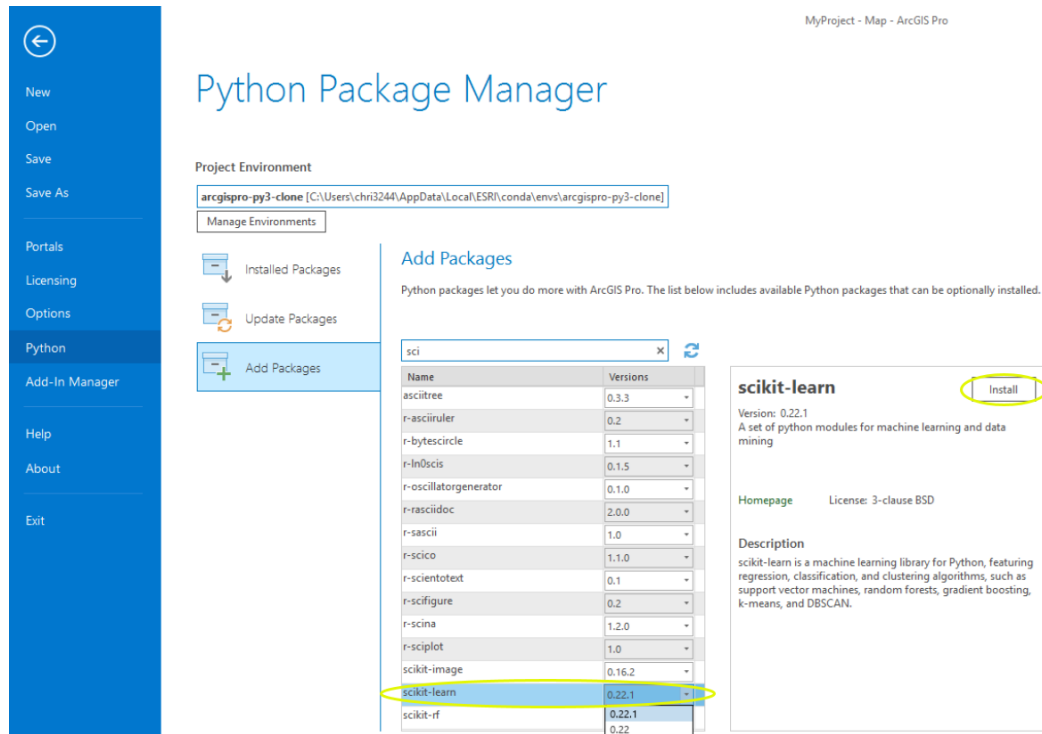
To run the WIM tools, users also must install the Scikit-Learn python package to their ArcGIS Pro python environment. Without doing so, tools “Train Random Trees,” “Run Random Trees,” and “Assess Accuracy” will fail. To install the Scikit-Learn package, users should:

1. Clone the default ArcGIS Pro python environment.
  - In an ArcGIS Pro project, select the “Project” tab
  - Select the “Python” tab
  - Select “Manage Environments”
  - Select “Clone Default”

### Python Package Manager



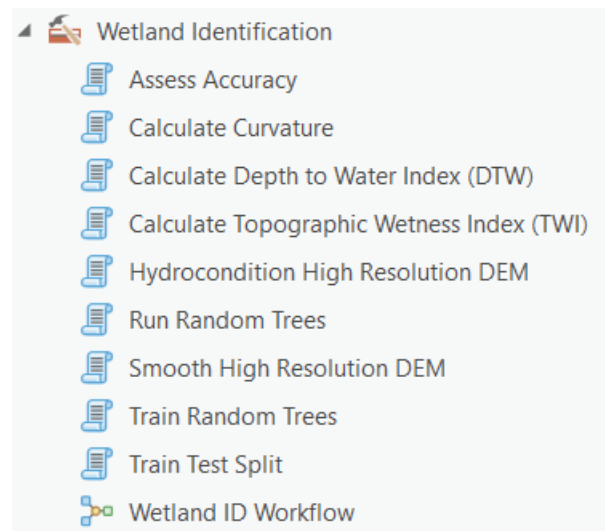
2. Add the Scikit-Learn package to the cloned environment.
  - With the cloned environment selected in the Python Package Manager page, select “Add Packages”
  - Navigate to Scikit-Learn and install the latest version.



3. Ensure that this cloned, modified python environment is selected for the current session before using the WIM tools.

### 3.0 Solution overview

The WIM is implemented as nine Arc Hydro python tools in a separate toolset, titled “Wetland Identification,” and added to the Arc Hydro Tools Python Toolbox for ArcGIS Pro (Figure 1). Included in the toolset is an ArcGIS model that automates the execution of the tools in proper sequence.



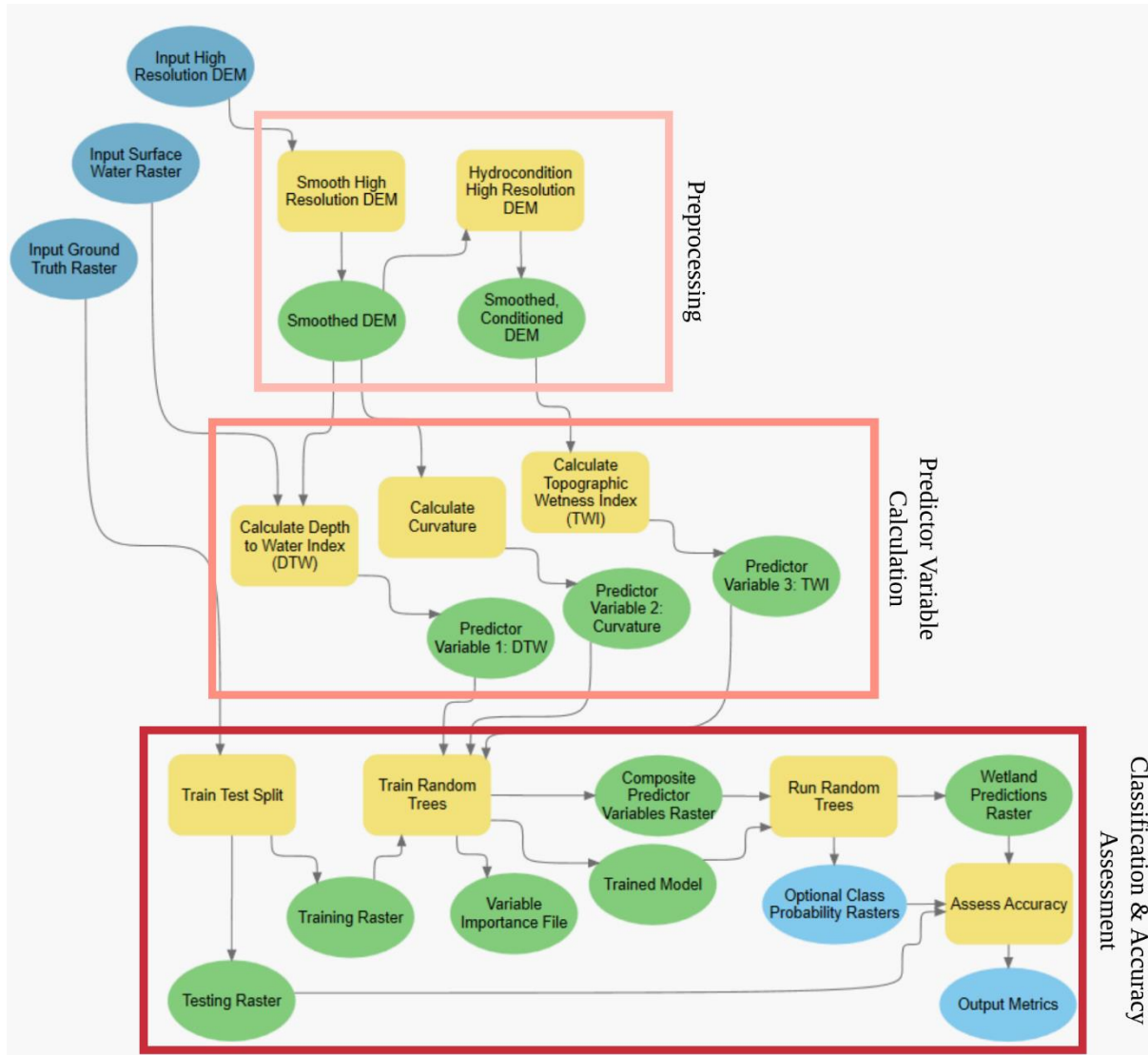
**Figure 1. Arc Hydro Pro toolset implementation of the Wetland Identification Model**

The WIM consist of three main parts: preprocessing, predictor variable calculation, and classification and accuracy assessment (Figure 2). Required input data are a high-resolution digital elevation model (DEM) and verified wetland/nonwetland coverage (i.e., ground truth data), both in TIFF format. The current implementation also requires a surface water input raster, although future implementations will derive this directly from the input DEM. An optional input layer is a training sampling constraint raster, explained in greater detail later in this document. Final model outputs are wetland predictions and an accuracy report. The current implementation of the WIM uses the TIFF raster format for output rasters. Executed as intended, the workflow (Figure 2) should be as follows.

1. The input DEM is smoothed and then conditioned.
2. The preprocessed DEM is used to calculate the predictor variables: the topographic wetness index (TWI), curvature, and cartographic depth-to-water index (DTW).
3. Training data are derived from the ground truth data.
4. The training data are coupled with the merged predictor variables to train a Random Trees (Breiman, 2001) model.



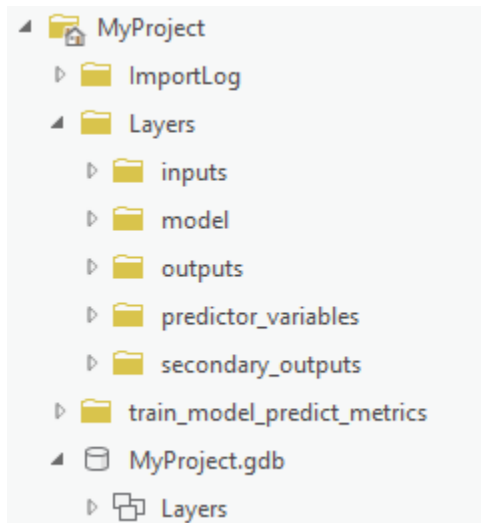
- The ground truth data that were not used to train the model are used to assess the accuracy of predictions.



**Figure 2. Overview of the Wetland Identification Model**

### 3.1 Database design

It is recommended to follow the presented database design. The folder structure is as follows.



Although the folder structure mirrors the Arc Hydro design, the Layers feature dataset is unused as no feature classes are created. The contents of the other folders are as follows.

“MyProject\Layers” subfolders:

1. “inputs”
  - a. Stores the input data in TIFF format. These data must include the DEM, ground truth dataset, and surface water raster. This folder will also optionally contain the training area constraint raster.
  - b. If running multiple trials of the WIM, the contents of this folder will not change.
2. “model”
  - a. Stores data that directly impact or are directly impacted by the Random Trees model. This includes the training and testing rasters, and the variable importance measures that are calculated each time a model is trained.
  - b. The contents of these files should be kept for reference over the course of trials that evaluate different training sampling scenarios.
3. “outputs”
  - a. Stores the prediction outputs from the Random Trees model. This will always include a prediction raster, where each cell is assigned a class. This folder may also include prediction probability rasters.

4. “predictor\_variables”

- a. Stores the predictor variables that are used to train the Random Trees model. If preprocessing parameters change, new contents will be added to this folder.

5. “secondary\_outputs”

- a. Stores intermediate raster outputs. These will include the smoothed DEM, hydroconditioned DEM, TWI components, and DTW components. If users have already used these rasters to derive predictor variables (see sections 2.3.3 – 2.3.5), the contents of this folder can be archived to free disk space.

“MyProject\train\_model\_predict\_metrics” contents:

1. This folder (named according to user input) contains the accuracy metrics used to summarize model performance. These are not GIS data, but rather various plots and tables.

## 3.2 Output data naming conventions

The WIM creates numerous output data structures. To ensure consistency of application and result data, resulting data should follow predetermined data and naming conventions. These include the following:

1. Processed DEMs are saved with compounding prefixes that describe the preprocessing methods applied. For example, “dem\_pm\_100\_fil” is a DEM that was smoothed using the Perona-Malik method with 100 iterations, then filled. This is recommended to keep track of the effect of DEM preprocessing on wetland modeling accuracy
2. By default, predictor variables are saved with the base name of the preprocessed DEM and a suffix with the pattern “\_[var].tif.” This is recommended to keep track of the effect of DEM preprocessing on wetland modeling accuracy, and how the best-performing preprocessing methods may differ across predictor variables.
3. Training and testing rasters are saved as “train.tif” and “test.tif” by default. However, users are encouraged to edit these output names to reflect the training sampling scenario applied, since the base name of the training raster is adopted by subsequent prediction outputs and the accuracy metrics directory. An example of recommended naming convention would be “train\_0\_50\_1\_20.tif” and “test\_0\_50\_1\_80.tif”. These names reflect that 50% of class 0 and 20% of class 1 were used for training, and the accuracy assessment will apply to the remaining 50% and 80% of class 0 and 1 cells, respectively.

4. As detailed above, prediction outputs and the location of accuracy metrics are named using the training raster base name by default. This is recommended to keep track of the effects of training sampling scenarios on model accuracy.

### 3.3 Tool operations

The following sections present operation for the nine WIM tools. Although we provide suggestions and starting points for the execution of the tools, users should keep in mind that WIM parameters will vary by the landscape and application. It will likely be an iterative process to find the WIM parameters best suited to a user's specific study area and end goal for results. For more detailed discussion on the methods applied and the justification for their use in the WIM workflow, see O'Neil et al. (2018; 2019).

#### 3.3.1 Preprocessing: *Smooth High-Resolution DEM*

This tool creates a smoothed DEM using one of four methods: mean, median, Gaussian, or Perona-Malik. These four methods were selected for their common use in hydrology-related applications. Smoothing is used to blur DEMs to remove the changes in elevation that are too small to indicate features of interest (i.e., microtopographic noise), which are ubiquitous in high-resolution DEMs. The scale of smoothing, or proxy for the scale of smoothing, determines the size of features that are preserved. In many cases, the Perona-Malik smoothing method with at least 50 smoothing iterations is a good starting point for analyses. However, users are encouraged to explore other options depending on the application. For more detailed discussion, users should see O'Neil et al. (2019). Inputs and outputs are shown in the example run.

The tool performs the following actions:

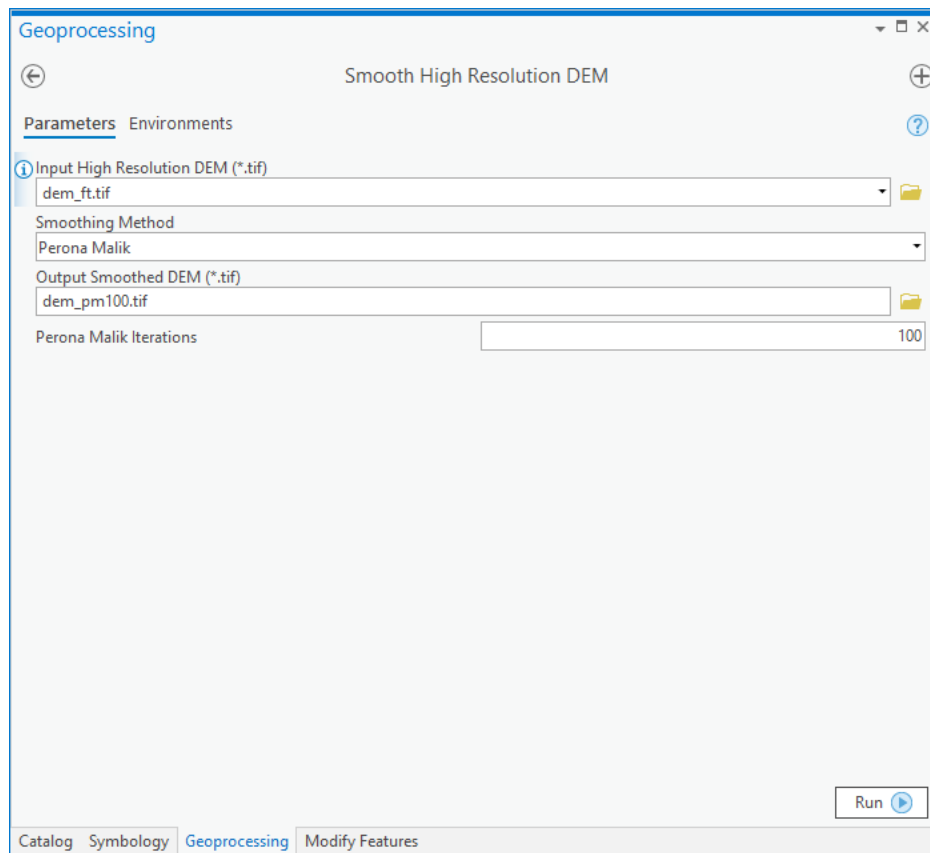
1. Applies a selected smoothing method
  - a. Mean smoothing performs a linear convolution using the user-defined smoothing width (meters). This smoothing window slides across the input DEM and the center cell of each window is replaced with the mean of all cells within the window. The rate of smoothing is determined by the smoothing width.
  - b. Median smoothing performs a nonlinear convolution using the user-defined smoothing width (meters). This smoothing window slides across the input DEM and the center cell of each window is replaced with the median of all cells within the window. Median smoothing is less affected by outliers in windows and is typically well-suited to salt-and-pepper type noise. The rate of smoothing is determined by the smoothing width.
  - c. Gaussian smoothing performs a linear convolution using a Gaussian function. The rate of smoothing is determined by the Gaussian kernel, where the standard deviation parameter acts as a proxy for a smoothing width (meters). **Users should note that Gaussian smoothing results in boundary effects that cause erroneous values at the edges of the input DEM. The smoothed DEM will be returned with smaller extents to remove these values.**

- d. Perona-Malik smoothing performs a nonlinear, anisotropic smoothing. This method estimates geomorphic feature boundaries to be where the slope is steeper than 90% of all slopes within the DEM. Smoothing is promoted only between estimated feature boundaries to preserve feature edges and achieve preferential smoothing. The rate of smoothing is determined by the number of smoothing iterations.

## 2. Applies a rate of smoothing parameter

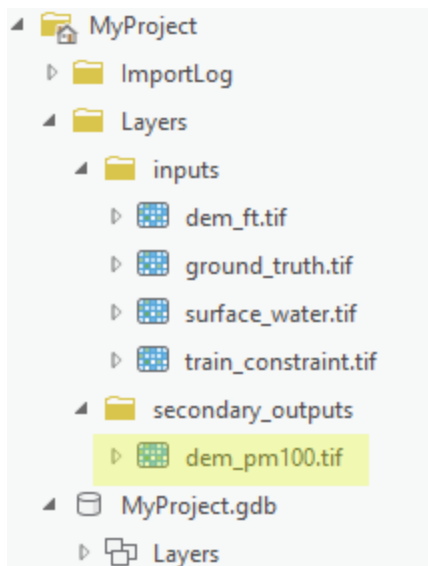
- a. For mean and median smoothing, the smoothing width refers to the X and Y length of the smoothing window in meters. If linear units of the DEM are not in meters, the conversion is executed automatically. Features smaller than the area of the smoothing window will be blurred.
- b. For Gaussian smoothing, the smoothing width is used to approximate the standard deviation used in the Gaussian kernel, according to Lashermes et al. (2007). If linear units of the DEM are not in meters, the conversion is executed automatically. Execution of Gaussian smoothing in the WIM is based on code from Sangireddy et al. (2016).
- c. For Perona-Malik smoothing, the number of iterations determines the scale of features preserved and smoothed, however this parameter has no unique and uniform equivalent spatial scale (Passalacqua, Do Trung, et al., 2010). A higher number of iterations will result in coarser output landscapes. Execution of Perona Malik smoothing in the WIM is based on code from Sangireddy et al. (2016).

### 3.3.1.1 Example run



### 3.3.1.2 Example output structure

The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.



### 3.3.2 Preprocessing: *Hydrocondition High-Resolution DEM*

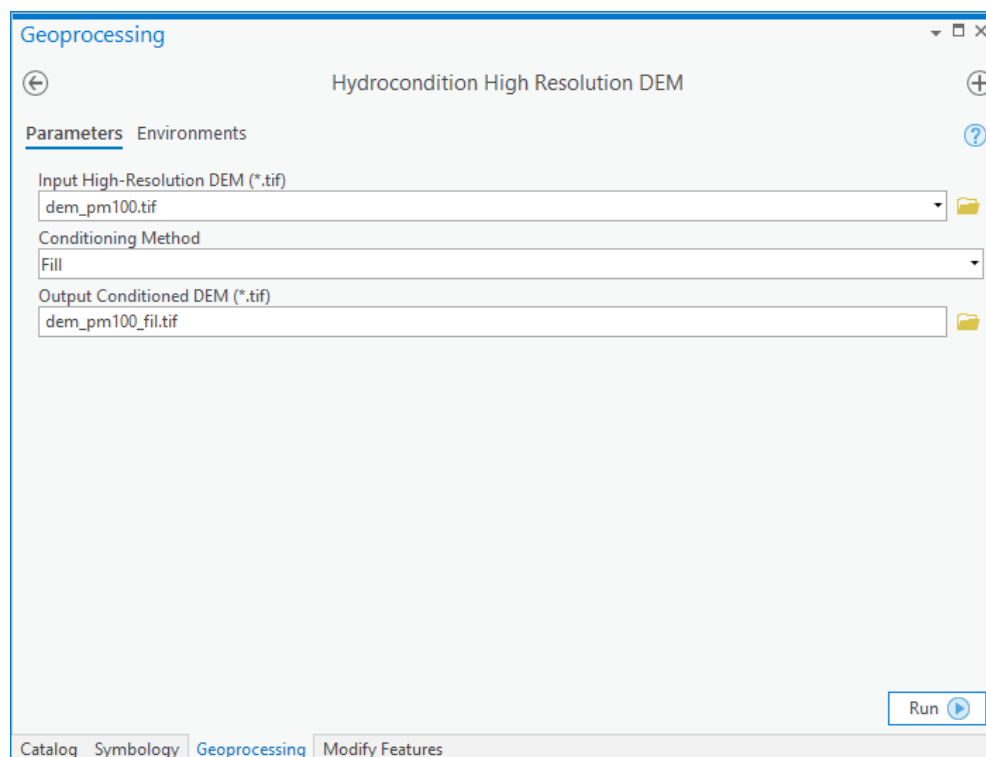
This tool resolves topographic depressions using Fill, a widely used method for extracting hydrologic parameters from DEMs. Topographic depressions may represent erroneous or actual features. If topographic depressions are not resolved, they can impact overland flow modeling by accumulating water and creating flow path discontinuities (Jenson & Domingue, 1988; O'Callaghan & Mark, 1984). The sensitivity of hydrologic modeling to the hydroconditioning technique chosen increases with the resolution of the input DEM, making this step especially important for hydrologic modeling from LiDAR DEMs (Grimaldi et al., 2007; Lindsay, 2016; Lindsay & Creed, 2005; Woodrow et al., 2016). In the current implementation, the only available method is Fill. However, future implementations may include methods that are better suited to hydrologically correcting high-resolution DEMs. Inputs and outputs are shown in the example run.

The tool performs the following actions:

1. Applies the Fill hydroconditioning method. Fill removes depressions by adjusting the elevation of a depression pixel to match the elevation of the surrounding pixels (Jenson & Domingue, 1988; Planchon & Darboux, 2002; Wang & Liu, 2007).

#### 3.3.2.1 Example run

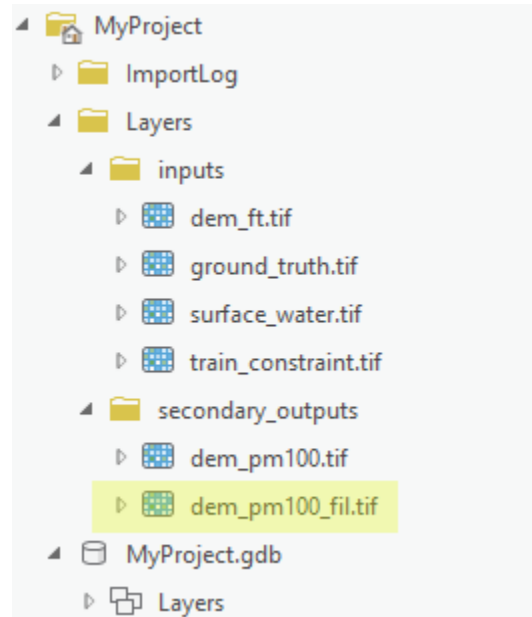
It is ***highly recommended*** that users select the smoothed high-resolution DEM as the input DEM, although the tool will also operate on regular DEMs.





### 3.3.2.2 Example output structure

The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.



### 3.3.3 Predictor Variable Calculation: *Calculate Depth to Water Index*

This tool calculates the cartographic depth-to-water index (DTW) to be subsequently used as a predictor of wetland areas. The DTW, developed by Murphy et al. (2007), is a soil moisture index based on the assumption that soils closer to surface water, in terms of distance and elevation, are more likely to be saturated. Calculated as a grid, the DTW is defined as

$$DTW (m) = \left[ \sum \left( \frac{dz_i}{dx_i} \right) a \right] * x_c, \quad (1)$$

where  $\frac{dz}{dx}$  is the downward slope of cell  $i$  along the least-cost (i.e., slope) path to the nearest surface water cell,  $a$  is a factor accounting for flow moving parallel or diagonal across pixel boundaries, and  $x_c$  is the cell resolution (Murphy et al., 2007). The WIM implementation of eq. (1) requires two inputs: a slope raster to represent the cost surface and a surface water raster to represent the source location. **In its current implementation, users must provide a surface water raster; however, this will become optional in the future when the tool has the capability to estimate open water locations directly from the DEM.** Inputs and outputs are shown in the example run.

The tool performs the following actions:

1. Calculates a DTW-specific slope raster from the input DEM, unless one is provided as an optional input.
2. Optionally saves the intermediate outputs created during processing.
3. Creates the DTW raster using the surface water raster as the source and the DTW slope raster as the cost.

### 3.3.3.1 Example run

It is ***highly recommended*** that users calculate the DTW from a smoothed, high-resolution DEM. Otherwise, microtopographic features will be modeled by the resulting DTW raster and may decrease the accuracy of the final wetland predictions. It is also ***unnecessary to use the hydroconditioned DEM*** here, since no flow paths are being generated. It is recommended to use just the smoothed DEM to provide a less-altered representation of the land surface. **Note, in this current implementation, the Input Surface Water Raster is required.**

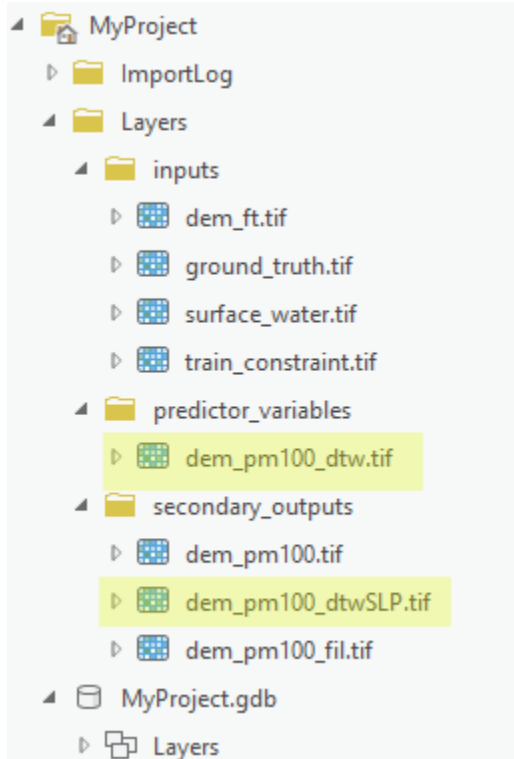
The screenshot shows the 'Calculate Depth to Water Index (DTW)' tool in the Geoprocessing environment. The interface includes a 'Parameters' tab and an 'Environments' tab. The 'Parameters' section contains the following fields and options:

- Input Smoothed DEM:** dem\_pm100.tif
- Output DTW Raster (\*.tif):** dem\_pm100\_dtw.tif
- ☒ **Save intermediate outputs**
- Output DTW Slope:** dem\_pm100\_dtwSLP.tif
- Pre-Calculated DTW Components (Optional):**
  - Input DTW Slope Raster:** (empty field)
  - Input Surface Water Raster:** surface\_water.tif

A 'Run' button is located at the bottom right of the tool window. The bottom of the window shows a tab bar with 'Catalog', 'Symbology', 'Geoprocessing' (selected), and 'Modify Features'.

### 3.3.3.2 Example output structure

The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.



### 3.3.4 Predictor Variable Calculation: *Calculate Curvature*

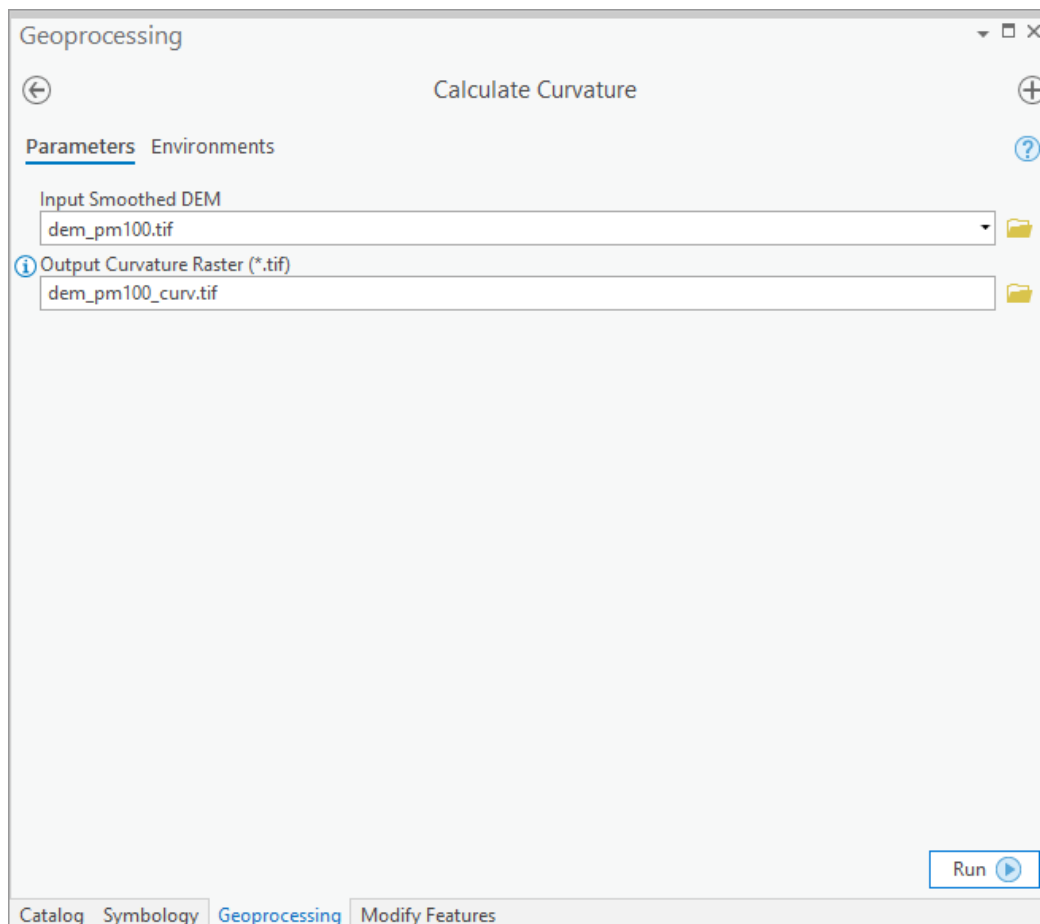
This tool calculates the curvature (i.e., the second derivative) of the land surface to be subsequently used as a predictor of wetland areas. Curvature can be used to describe the degree of convergence and acceleration of flow (Moore et al., 1991). Inputs and outputs are shown in the example run.

The tool performs the following actions:

1. Calculates the curvature of the input.

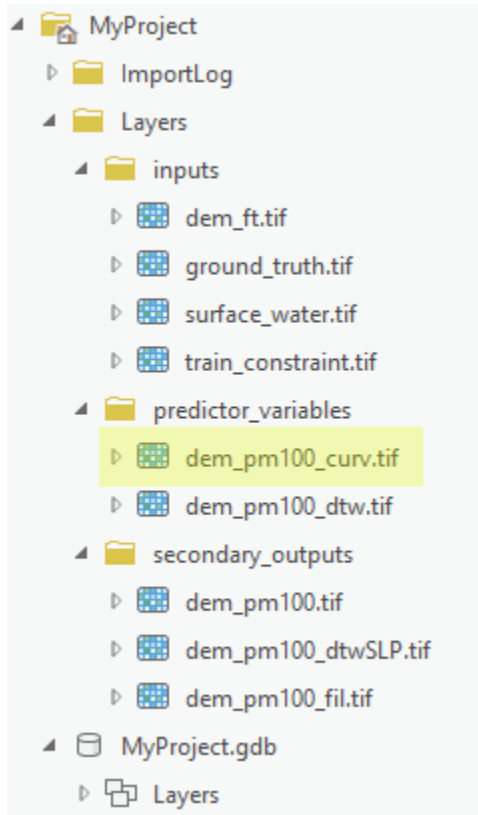
#### 3.3.4.1 Example run

It is ***highly recommended*** that users calculate curvature from a smoothed, high-resolution DEM. Otherwise, microtopographic features will be modeled by the resulting curvature raster and may decrease the accuracy of the final wetland predictions. It is also ***unnecessary to use the hydroconditioned DEM*** here, since no flow paths are being generated. It is recommended to use just the smoothed DEM to provide a least-altered representation of the land surface.



### 3.3.4.2 Example output structure

The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.



### 3.3.5 Predictor Variable Calculation: *Calculate Topographic Wetness Index*

This tool calculates the topographic wetness index (TWI) to be subsequently used as a predictor of wetland areas. The TWI relates the tendency of an area to receive water to its tendency to drain water, and is defined as

$$TWI = \ln\left(\frac{\alpha}{\tan \beta}\right), \quad (2)$$

where  $\alpha$  is the specific catchment area (contributing area per unit contour length) and  $\tan(\beta)$  is the local slope (Beven & Kirkby, 1979). The WIM implementation of eq (2) requires a TWI slope and specific catchment area as inputs, although both can be calculated directly from the input high-resolution DEM. Inputs and outputs are shown in the example run.

The tool performs the following actions:

1. Calculates a TWI-specific slope raster from the input DEM, unless one is provided as an optional input.
2. Calculates a specific catchment area raster from the input DEM, unless one is provided as an optional input.
3. Optionally saves the intermediate outputs created during processing.
4. Creates the TWI raster by implementing eq. (2) as a raster algebra expression.

### 3.3.5.1 Example run

It is ***highly recommended*** that the input DEM be the result of the Smooth High-Resolution DEM tool followed by the Hydrocondition High-Resolution DEM tool (i.e., a smoothed, conditioned high-resolution DEM). Users must use a hydroconditioned input DEM to avoid undefined values in the resulting TWI raster. In addition, without smoothing microtopographic features may affect the accuracy of the final wetland predictions.

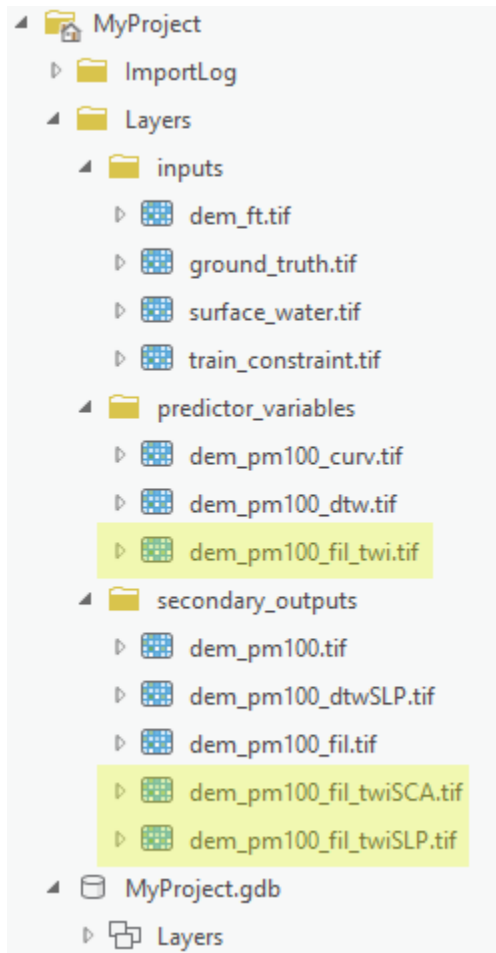
The screenshot shows the 'Geoprocessing' window with the 'Calculate Topographic Wetness Index (TWI)' tool selected. The 'Parameters' tab is active, displaying the following settings:

- Input Hydroconditioned DEM:** dem\_pm100\_fil.tif
- Output TWI Raster (\*.tif):** dem\_pm100\_fil\_twi.tif
- Save Intermediate Rasters:** ☒
- Output TWI Slope (\*.tif):** dem\_pm100\_fil\_twiSLP.tif
- Output TWI Specific Catchment Area (\*.tif):** dem\_pm100\_fil\_twiSCA.tif
- Pre-Calculated TWI Components (Optional):**
  - Input TWI Slope Raster:** (empty dropdown)
  - Input TWI Specific Catchment Area Raster:** (empty dropdown)

A 'Run' button with a play icon is located at the bottom right of the tool's parameter area. At the bottom of the window, a tab bar shows 'Catalog', 'Symbology', 'Geoprocessing' (selected), and 'Modify Features'.

### 3.3.5.2 Example output structure

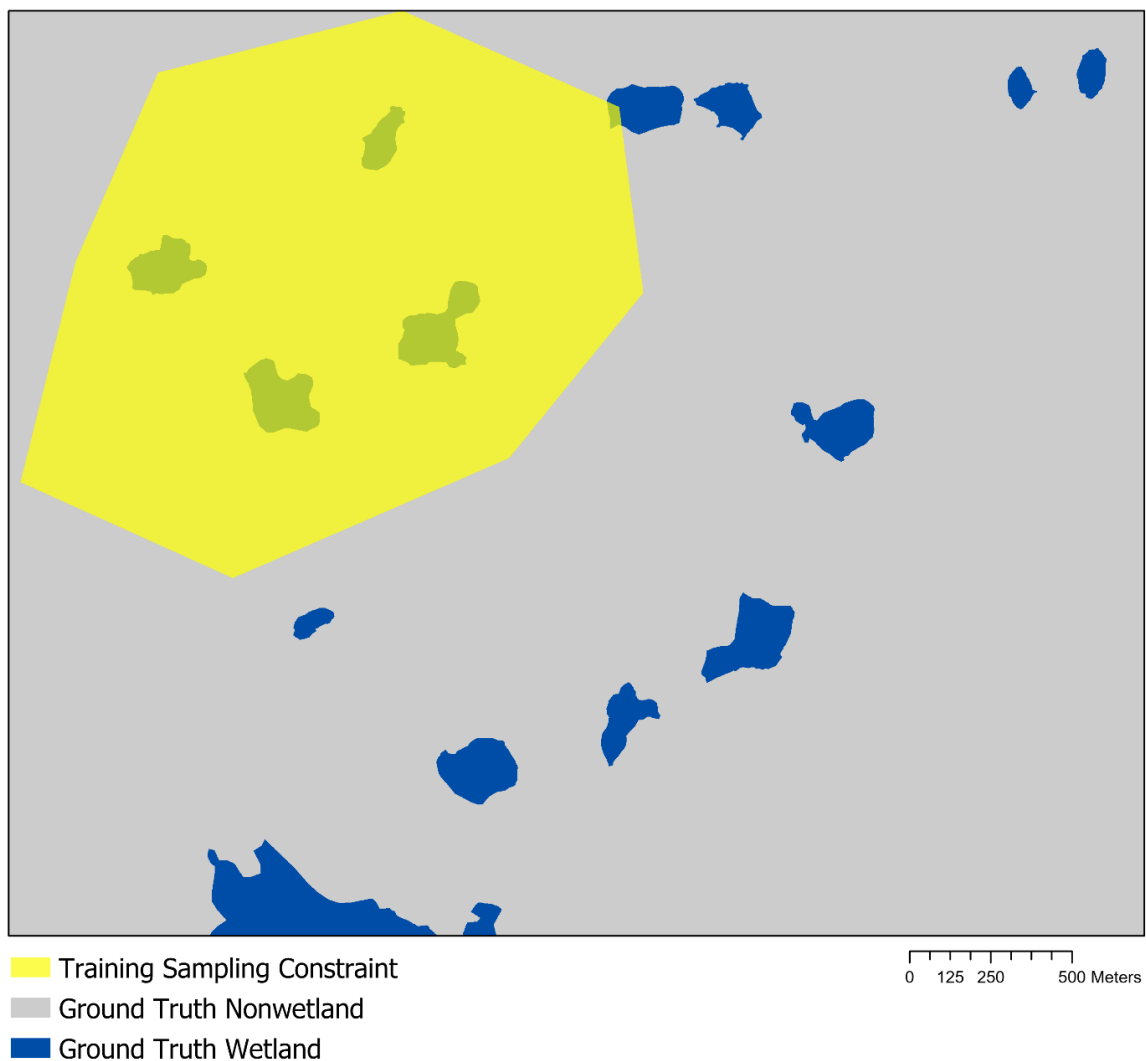
The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.





### 3.3.6 Classification and Accuracy Assessment: *Train Test Split*

This tool splits the input ground truth raster into two subsets: training and testing data. The training raster is created by randomly sampling a user-defined percentage of each ground truth class. The testing raster is the complement of the training raster, comprised of the remaining, unsampled cells. The training raster is subsequently used to train the supervised classification algorithm by providing cells labeled with their true class. In a later step, these are matched to predictor variables for the same locations and used to learn the characteristics of that class. The testing raster is subsequently used to measure the accuracy of class predictions for cells that were not included in the training phase. For further discussion on training sampling scenarios for the WIM, see O’Neil et al. (2019). Inputs and outputs are shown in the example run. Below is an example of the inputs to this tool used for this demo.

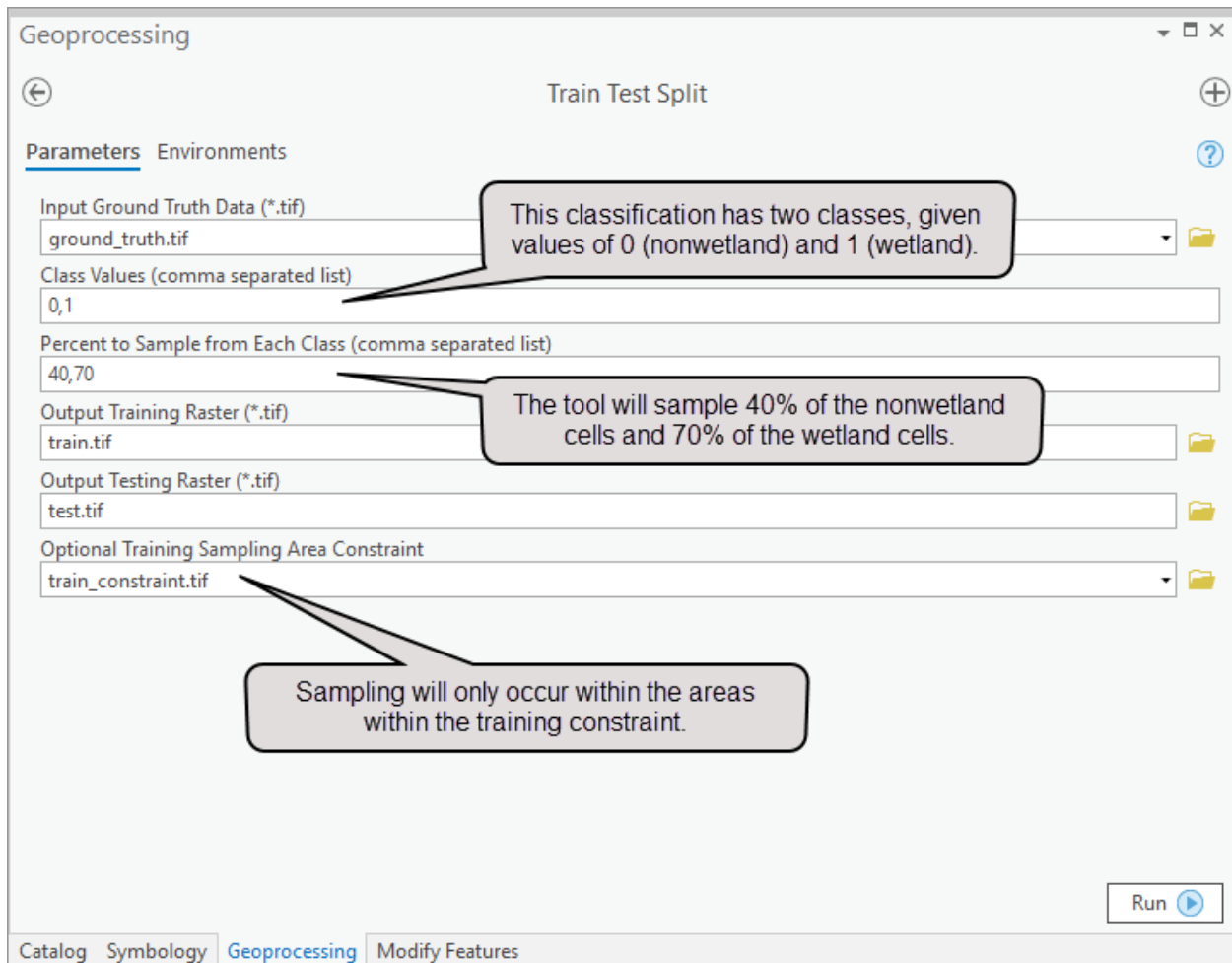


**Figure 3. Example of the ground truth dataset (grey and blue areas), which gives the true land cover, and a training sampling constraint, which delimits the portion of the ground truth dataset that training data can be sampled from.**

The tool performs the following actions:

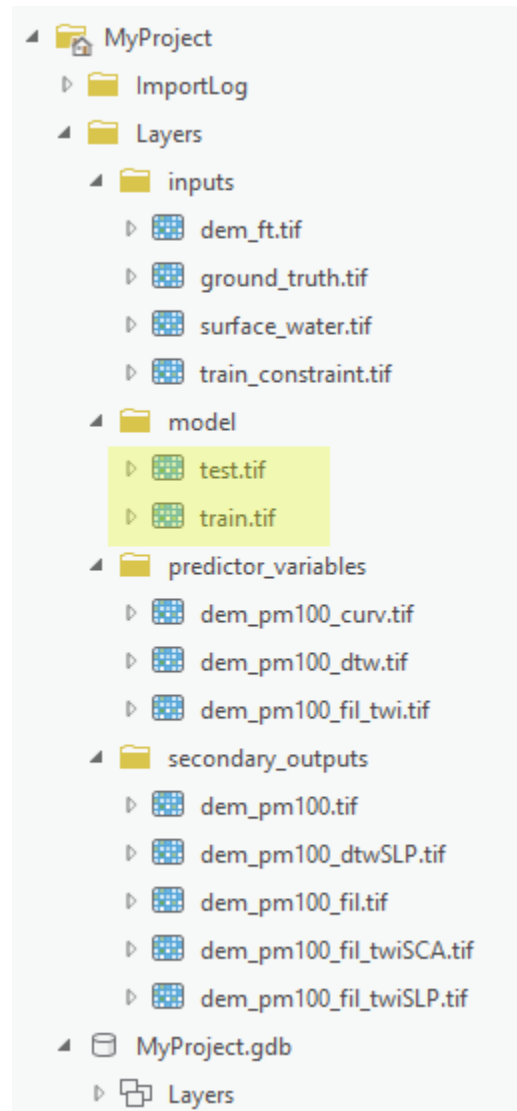
1. Checks that the input class names match exactly the unique values in the ground truth raster.
2. Checks if cells should be sampled from the entire extents of each class or only within the training sampling constraint.
3. Randomly samples the number of cells from each class that reflects the user-input percentage.
4. Combines the cells sampled from each class into a single raster and saves to the output training raster. The remaining cells are combined into another raster and saved to the output testing raster.

### 3.3.6.1 Example run



### 3.3.6.2 Example output structure

The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.



### 3.3.7 Classification and Accuracy Assessment: *Train Random Trees*

This tool executes the training phase of the Random Trees algorithm. In this phase, the algorithm takes bootstrap samples of the training dataset, including the labeled cells in the training raster and the predictor variables. A decision tree is created from each bootstrap sample, and all are used to learn the indicators of the ground truth classes based on information from the predictor variables. In doing this, the Random Trees algorithm is less susceptible to overfitting. This tool uses the Scikit-Learn Python library (Scikit-Learn Developers, 2017a) to execute the Random Trees algorithm. For a more detailed discussion of the algorithm and its fit for the WIM, see O'Neil et al. (2019). Inputs and outputs are shown in the example run.

The tool performs the following actions:

1. Prepares the predictor variable raster(s) for use by the Scikit-Learn library. If more than one predictor variable raster is given, a composite raster is created from all input predictor variables and saved to the Prepared Predictor Variable Raster name. If only one raster is passed, it will be copied and saved to the Prepared Predictor Variable Raster name, but it can be disregarded for later steps. In either case, the cells in the prepared raster are extracted only where the training labels before further use.
2. Initializes the Random Trees model according to the number of trees, the maximum tree depth, the maximum number of features, and the class weights. If users omit these parameters or leave them unaltered, default values are used. Users should see Scikit-Learn documentation for further details on these parameters (Scikit-Learn Developers, 2017a).
3. Trains the initialized model given the training dataset. Saves the trained model to a JOBLIB file.
4. Saves the variable importance measures to a TXT file. These measures provide an estimate of the decrease in accuracy of the model if that predictor variable was removed.

### 3.3.7.1 Example run

Geoprocessing

Train Random Trees

Parameters Environments

Input Training Raster (\*.tif)  
train.tif

Input Predictor Variables Raster (\*.tif) (v)  
dem\_pm100\_curv.tif  
dem\_pm100\_dtw.tif  
dem\_pm100\_fil\_twi.tif

Output Trained Model (\*.JOBLIB)  
D:\Data\WIM\_documentation\MyProject\Layers\model\train\_model.JOBLIB

Prepared Predictor Variable Raster (\*.tif)  
composite.tif

Output Variable Importance (\*.txt)  
D:\Data\WIM\_documentation\MyProject\Layers\model\train\_varimp.txt

Advanced Random Trees Parameters (Optional)

Number of Trees 100

Maximum Tree Depth  
None

Maximum Number of Features  
auto

Class Weights

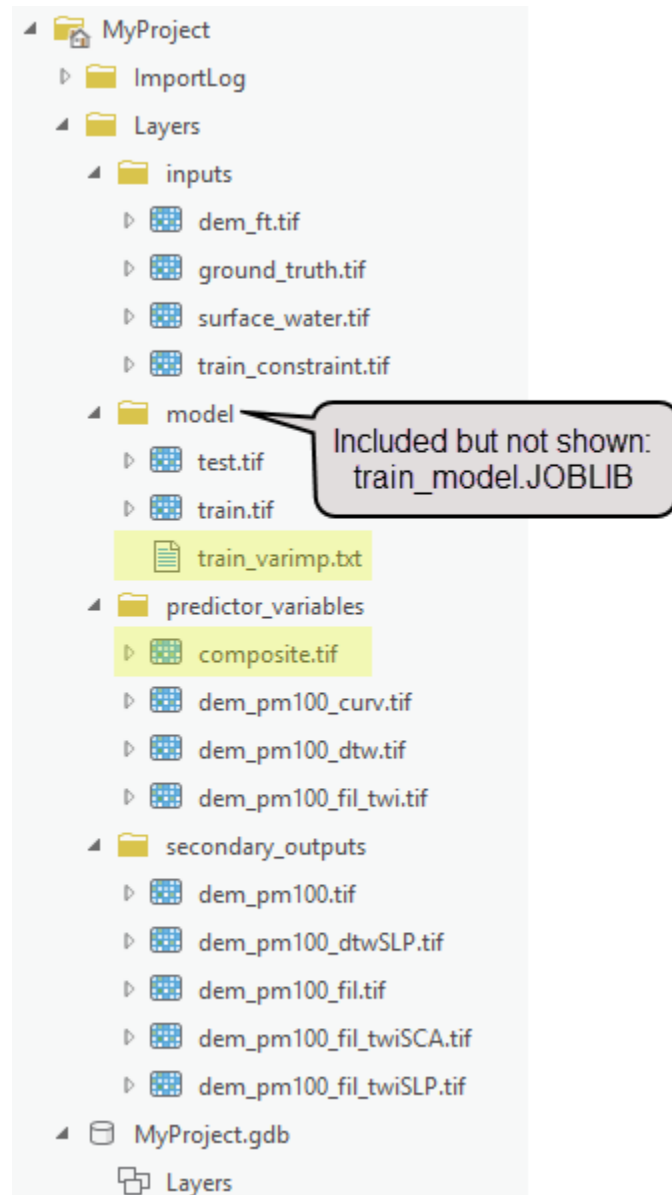
Target Class	Class Weight

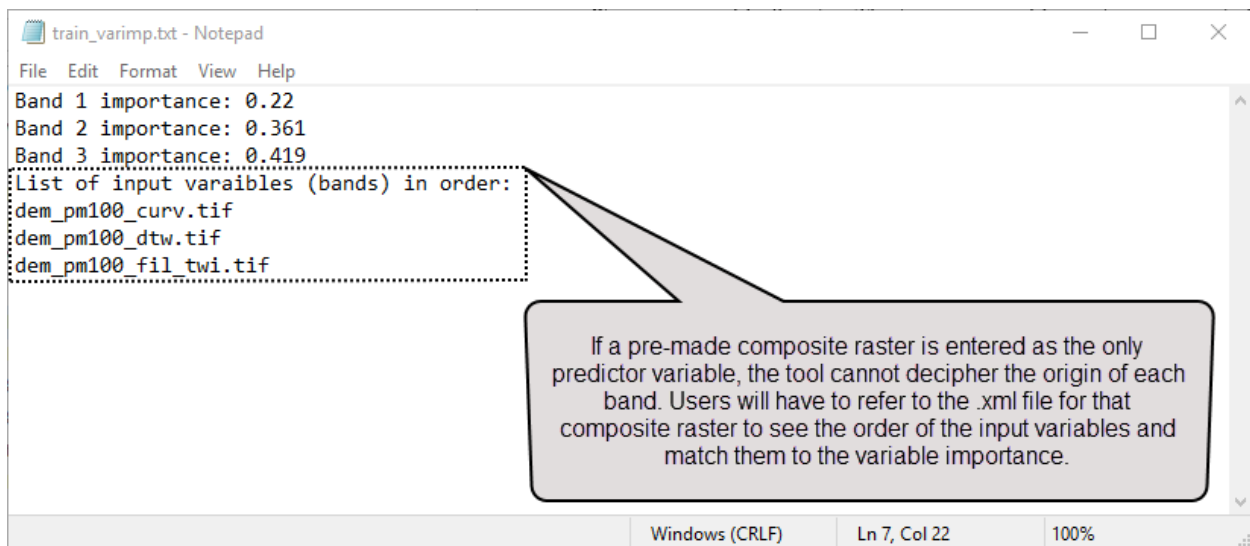
Run

Catalog Symbology Geoprocessing Modify Features

### 3.3.7.2 Example output structure

The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.





### 3.3.8 Classification and Accuracy Assessment: *Run Random Trees*

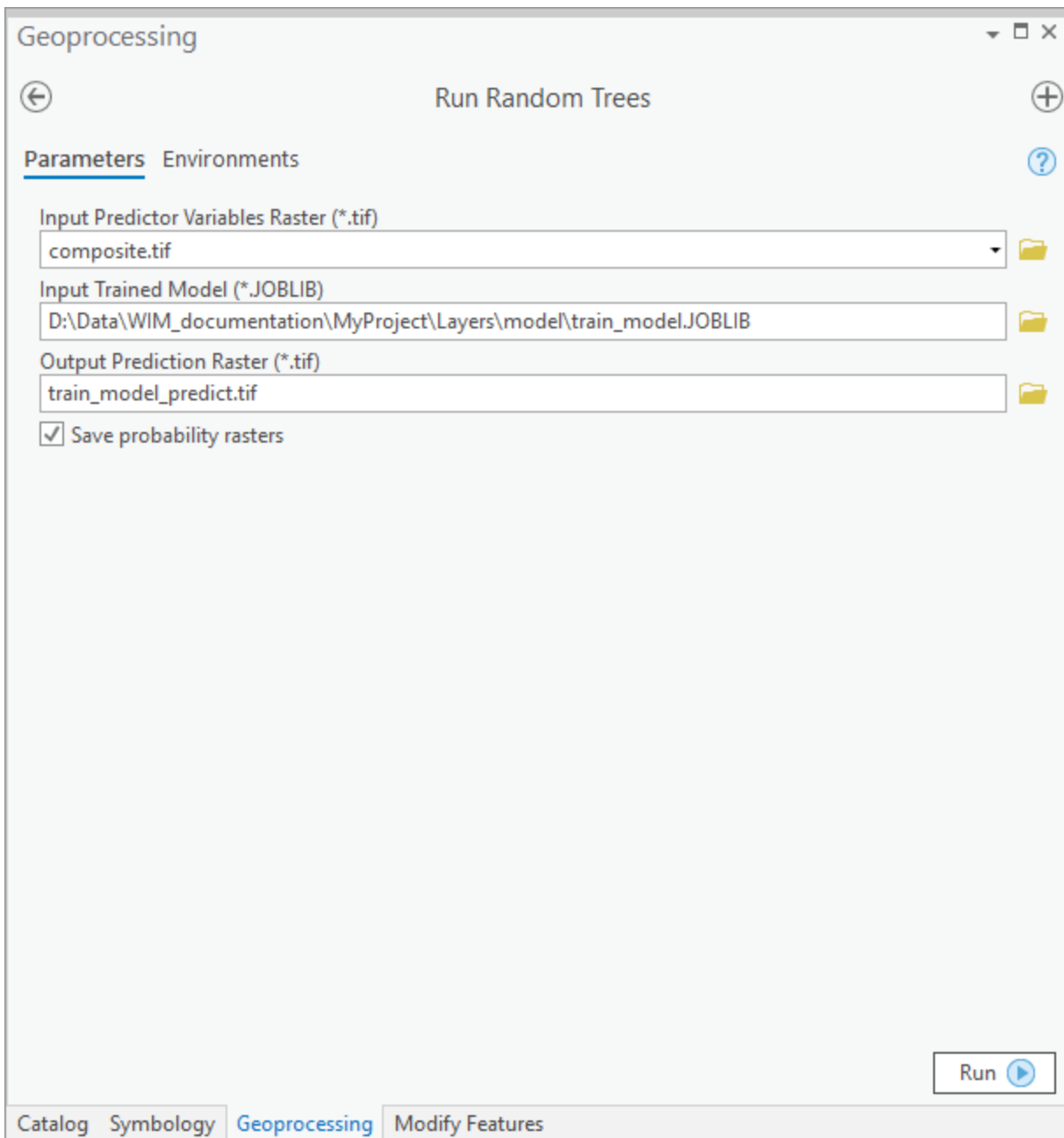
This tool uses the trained Random Trees model to predict the class for the cells of the input predictor variables. Predictions do not need to be made in the same area for which the model was trained, but the calculation of the predictor variable(s) must be the same. In this case, a model can be trained using the DTW, Curvature, and TWI for one area and used to make predictions for a new area, if DTW, Curvature, and TWI have also been derived and are used as inputs. Inputs and outputs are shown in the example run.

The tool performs the following actions:

1. Loads the JOBLIB model and uses it to predict wetland and nonwetland areas for the predictor variables raster. Internally, the random trees algorithm uses the relationships learned between the predictor variables and class values during training and determines the final predicted class based on the majority vote of all decision trees created.
2. If “Save probability rasters” is True, uses the JOBLIB model to produce the probability raster for each target class (e.g., wetland and nonwetland). The values in these rasters represent the probability that the cell belongs to the class in question on a 0-1 scale. These outputs can be useful for decision makers where the tradeoff between wetland detection and overprediction can be examined. Producing and saving these outputs also allow for a more thorough accuracy assessment in later steps.

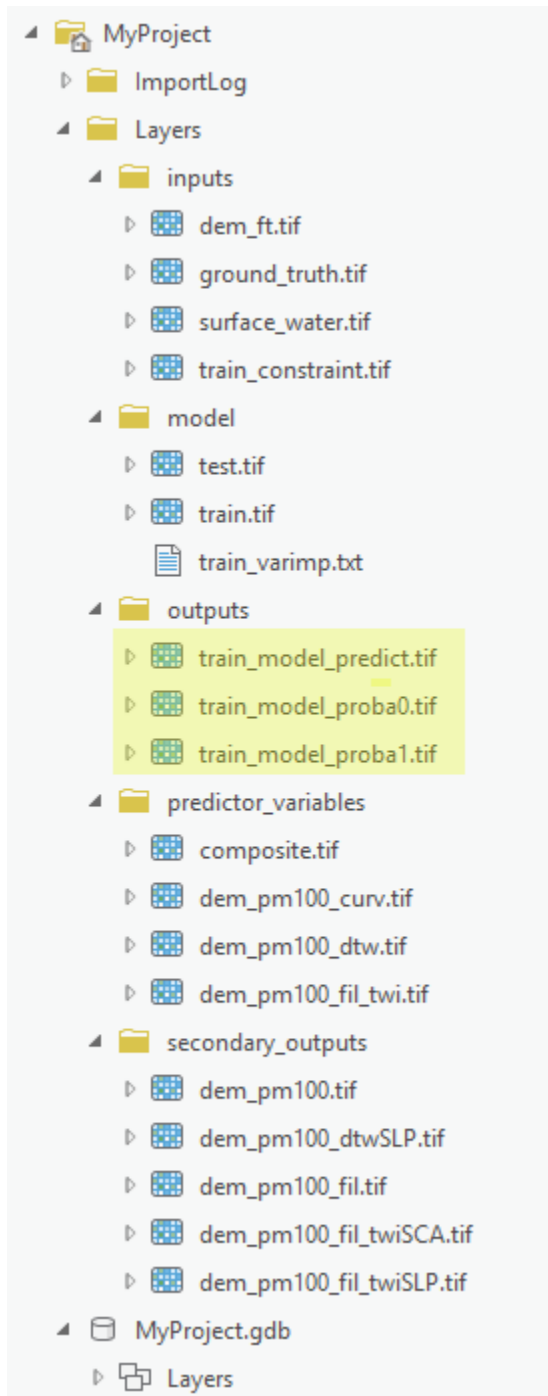


### 3.3.8.1 Example run

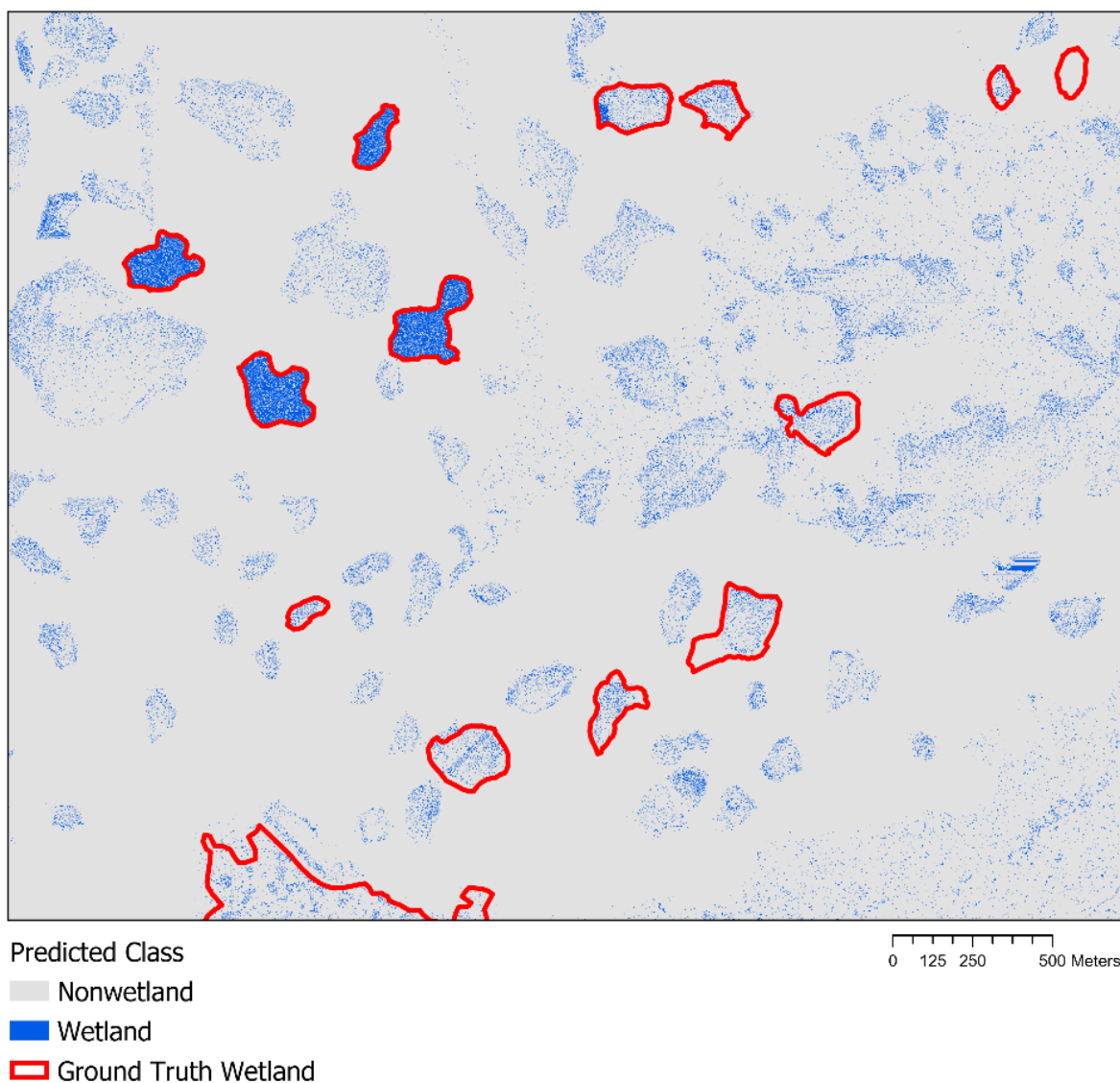


### 3.3.8.2 Example output structure

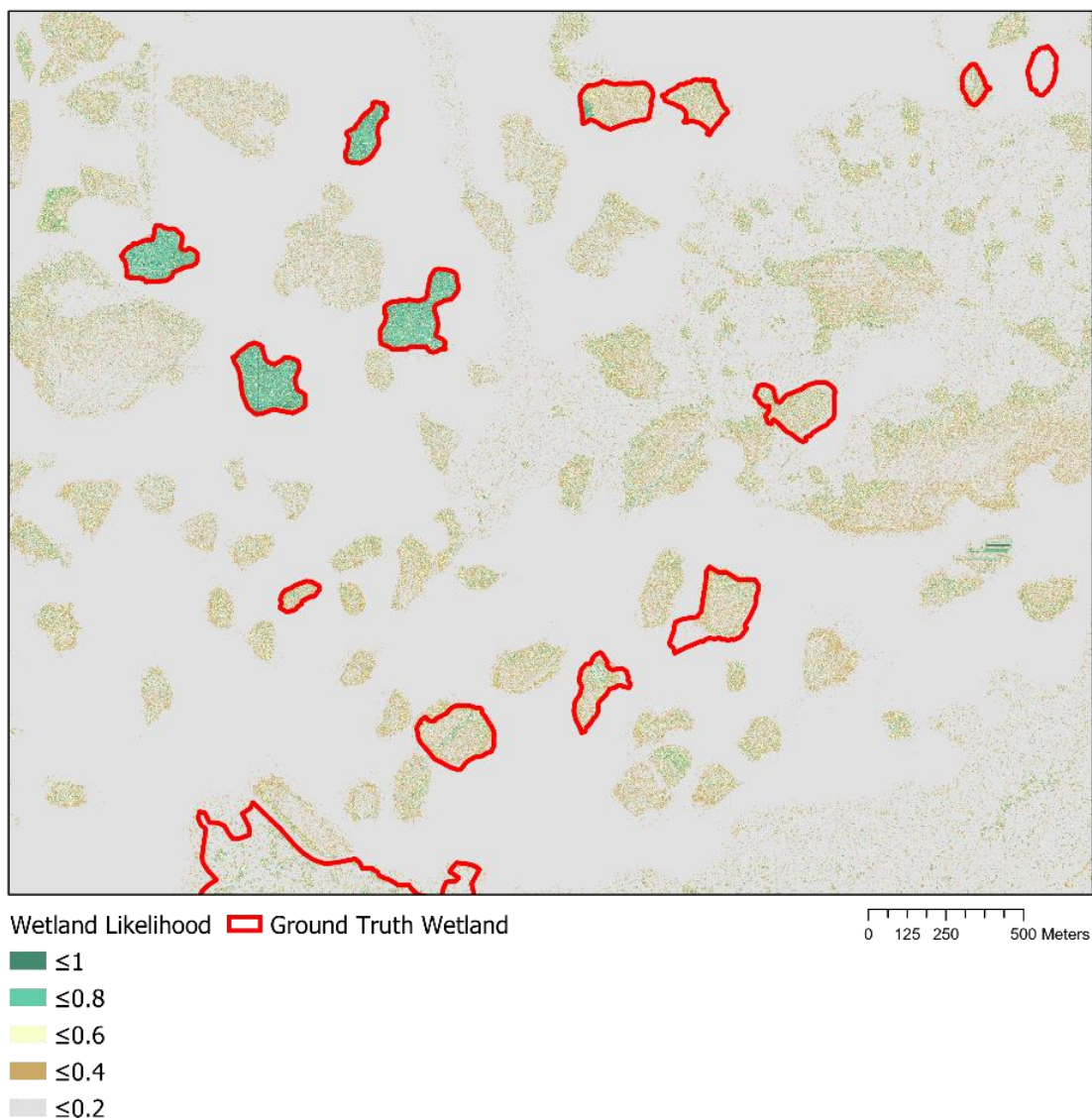
The following data structures were created as a result of the tool run (highlighted in yellow). Any output directories that do not exist will be created.



Below are images of model output created using the parameters outlined here. First, the model prediction, followed by the wetland class probability raster.



**Figure 4.** An example of wetland and nonwetland predictions produced by the workflow, with the outline of the ground truth wetlands shown for reference.



**Figure 5.** An example of the wetland likelihood estimated by the workflow, with the outline of the ground truth wetlands shown for reference.

### 3.3.9 Classification and Accuracy Assessment: Assess Accuracy

Given the predicted output from the preceding step and a corresponding raster with ground truth labels for the cells (i.e., the testing raster), this tool generates accuracy metrics that summarize the model's ability to predict wetlands and nonwetlands. Additional metrics are generated if the user includes output class probability rasters. Accuracy metrics were chosen to avoid misleading assessments of imbalanced ground truth classes, which is typical of wetland/nonwetland distributions. For further details and justification for the metrics chosen, see O'Neil et al. (2019). Accuracy metrics are calculated using the Scikit-Learn library (Scikit-learn Developers, 2017b). Inputs and outputs are shown in the example run. *Note that the accuracy assessment process assumes that every cell in the testing dataset has been assigned its ground truth class. That is, all wetland and nonwetland areas indicated are exhaustive of all wetland and nonwetland areas within the extents of the ground truth dataset. In the demo applications shown here, that is not necessarily true, and therefore, user should interpret these results as a demonstration of tool outputs rather than an indication of model performance.*

The tool performs the following actions:

1. Creates a new directory for accuracy metrics if the specified one does not already exist.
2. If necessary, extracts the prediction raster cells that overlap with the testing raster cells.
3. Calculates and plots a confusion matrix. The confusion matrix categorizes each cell (represented in units of km<sup>2</sup> and m<sup>2</sup>) into one of four groups: true positive, true negative, false positive, or false negative. These categories indicate that the predicted cell either correctly identified wetland area, correctly identified nonwetland area, incorrectly identified a wetland area, or incorrectly identified a nonwetland area, respectively. The plots are saved as PNG files to the accuracy metrics directory as "conf\_matrix" and "conf\_matrix\_meters."
4. Creates a classification report and saves to a TXT file as "summary\_stats." For each class, this report gives the precision, recall, f1-score, and support. Although metrics are calculated for each class, the descriptions below focus on the interpretation of scores for the wetland class. Note that other metrics given by the classification report may be misleading for imbalanced class predictions. For more information on these, users should see the Scikit-Learn documentation (Scikit-learn Developers, 2017b).

Precision is a metric that accounts for overprediction of the positive class (i.e., wetlands), without being biased by disproportionately large populations of the negative class (i.e., nonwetlands). Precision is the percentage of wetland predictions made that were correct, calculated as:

$$Precision = \frac{True\ Positive\ Predictions}{All\ Positive\ Predictions}. \quad (3)$$

Recall is a metric of class detection, giving the percentage of true wetlands that were correctly identified:

$$Recall = \frac{True\ Positive\ Predictions}{All\ True\ Wetlands}. \quad (4)$$

F1 score represents a weighted average of precision and recall, where the best F1 score is a value of 1 and a worst score is a value of 0. It is important to note that this metric assumes that high detection rate and low overprediction are equally important to stakeholders. Other forms of the F1 score exist where these components can be given different weights. The F1 score is defined as

$$F1 = \frac{2*(precision*recall)}{(precision+recall)}. \quad (5)$$

Finally, support represents the total samples (number of cells) in the true wetland class.

5. If class probability rasters are passed, calculates and plots precision-recall curves for each probability raster, which each corresponds to a single class, given. Precision-recall curves plot precision versus recall for each predictive threshold of that class. That is, the curve will show the precision and recall scores if each cell required 0-100% probability of belonging to a class before being assigned to that class. In addition, the Average Precision score is calculated for each precision-recall curve. Average Precision is a surrogate for the area under the precision-recall curve, and it is used to summarize model performance for the class of interest. An Average Precision score closer to 1 indicates a better performing model. The Average Precision metrics is calculated as,

$$Average\ Precision = \sum_n (R_n - R_{n-1})P_n, \quad (6)$$

where  $P_n$  and  $R_n$  are the precision and recall at the  $n^{th}$  predictive threshold, respectively.

### 3.3.9.1 Example run

Geoprocessing Assess Accuracy

[Parameters](#) [Environments](#) [?](#)

Input Testing Raster (\*.tif)  
test.tif

Input Prediction Raster (\*.tif)  
train\_model\_predict.tif

Output Metrics Directory  
train\_model\_predict\_metrics

Optional Accuracy Outputs

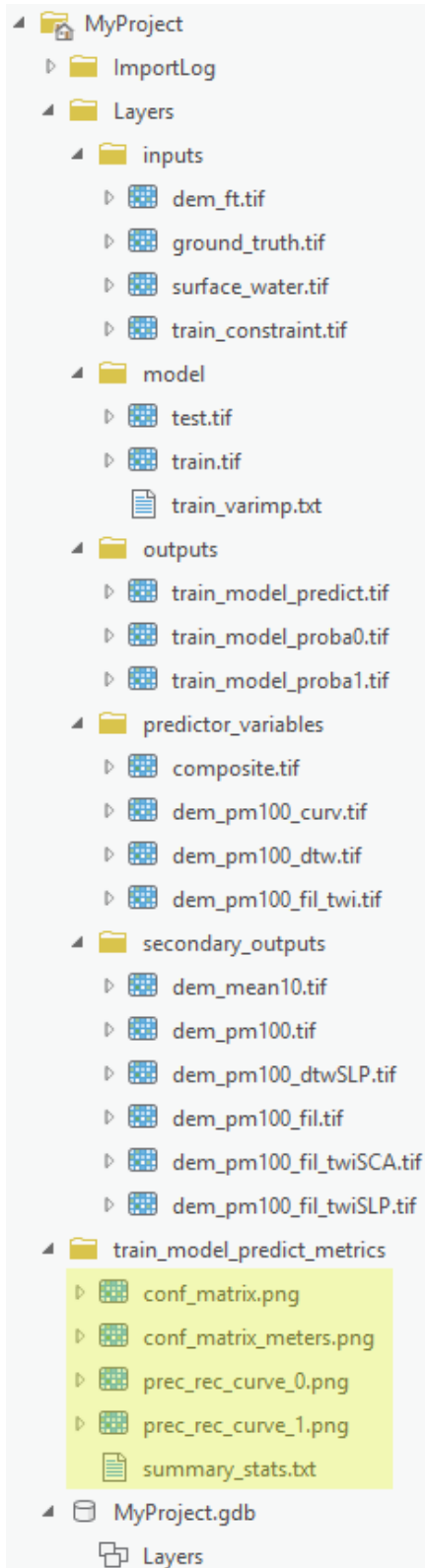
Target Class Value	Class Probability Raster (*.tif)
0	train_model_proba0.tif
1	train_model_proba1.tif

Run

Catalog Symbology **Geoprocessing** Modify Features

### 3.3.9.2 Example output structure

The following data structures were created as a result of the tool run (highlighted in yellow).





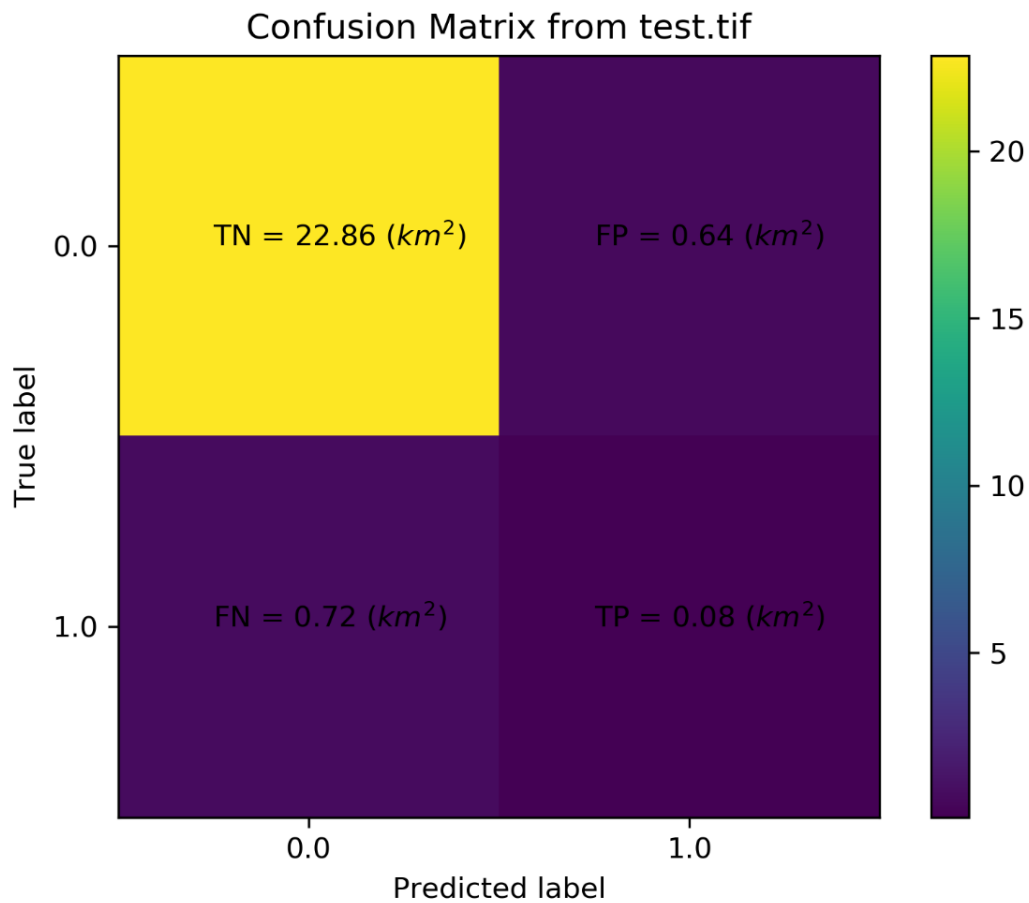


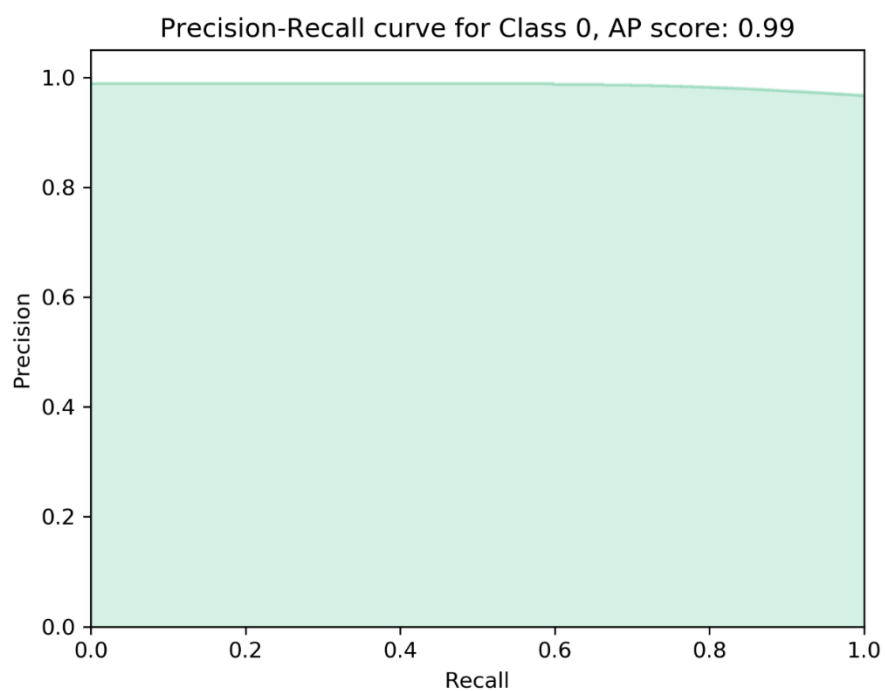
Figure 6. The confusion matrix generated by the accuracy assessment for the demo wetland identification application.

summary\_stats.txt - Notepad

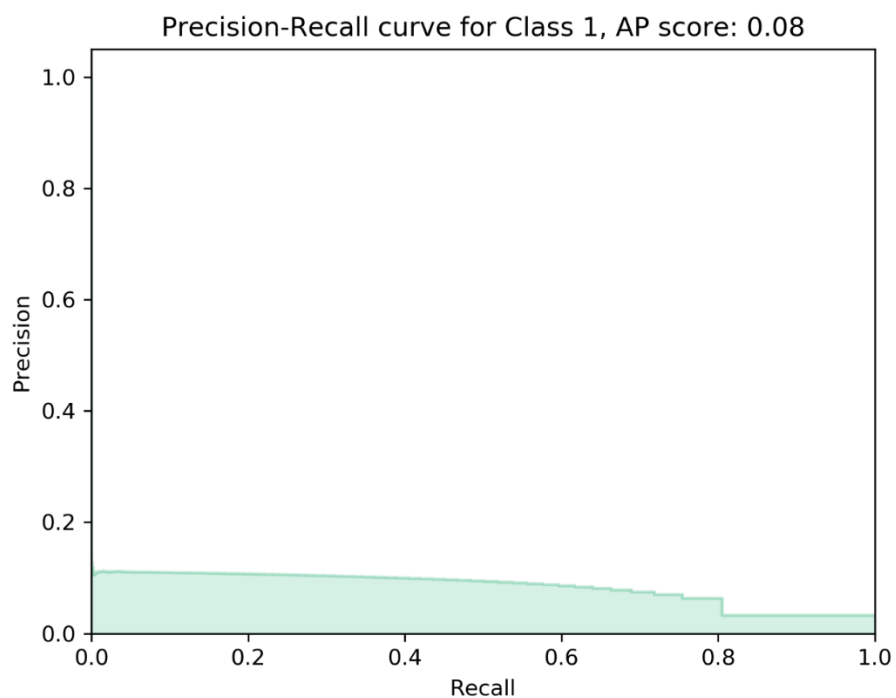
	precision	recall	f1-score	support
0.0	0.970	0.973	0.971	40484150
1.0	0.110	0.100	0.105	1370201
accuracy			0.944	41854351
macro avg	0.540	0.536	0.538	41854351
weighted avg	0.941	0.944	0.943	41854351

Windows (CRLF) Ln 1, Col 1 100%

Figure 7. The summary statistics report generated by the accuracy assessment for the demo wetland identification application.



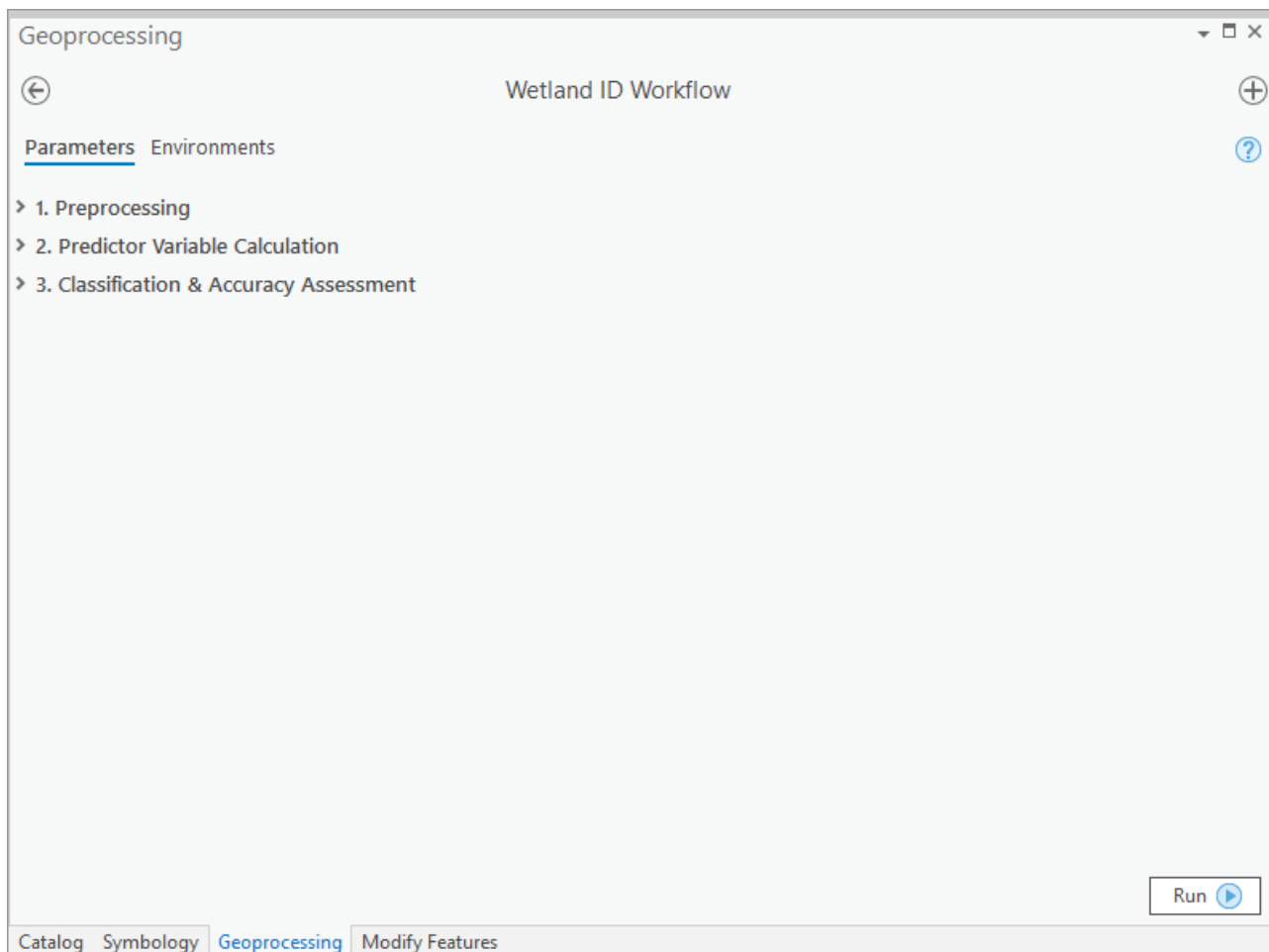
**Figure 8. The precision-recall curve for the 0-class predictions (nonwetland class) generated by the accuracy assessment for the demo wetland identification application.**



**Figure 9. The precision-recall curve for the 1-class predictions (wetland class) generated by the accuracy assessment for the demo wetland identification application.**

### 3.3.10 WIM tools as an ArcGIS Pro model

This model automates the entire WIM workflow by linking the above tools in proper sequence. As shown below, the tools are organized according to the workflow phase they fall under. Users must expand these workflow phase titles to input parameters. Executing this model should be considered as just the initial step in the process of developing one's own wetland model that is unique to their needs. After this initial step, users will find it helpful to iterate through tools such as Train Test Split or Smooth High-Resolution DEM in order to improve results and calibrate a model to a specific landscape. Where appropriate, users should complete this testing by iterating through the individual tools rather than executing this ArcGIS Pro model multiple times. Note that due to limitations in modifying the User Interface, users will need to re-run Assess Accuracy outside of the model to include probability rasters as inputs. Please keep in mind the [accuracy disclaimer](#).



### 3.3.10.1 Example run

In the demo below, we execute the workflow with following parameters, reflected in the output names.

- Mean smoothing with a 100 m smoothing width
- Fill hydroconditioning
- 10% of cells belonging to class 0 (nonwetlands) and 60% of cells belonging to class 1 (wetlands) sampled for training data. Note, sampling of these cells is restricted by the input training constraint.
- 100 decision trees grown
- Class weights of 1 and 50 for nonwetland and wetland predictions, respectively.
- Specifying this training/model application to be labeled as “demo”.

Geoprocessing

Wetland ID Workflow

Parameters Environments

1. Preprocessing

Input High Resolution DEM (\*.tif)  
dem\_ft.tif

Output Smoothed DEM (\*.tif)  
dem\_mean100.tif

Output Conditioned DEM (\*.tif)  
dem\_mean100\_fil.tif

Smoothing Method  
Mean

Smoothing Width (m)  
100

Smoothing Iterations  
100

Hydroconditioning Method  
Fill

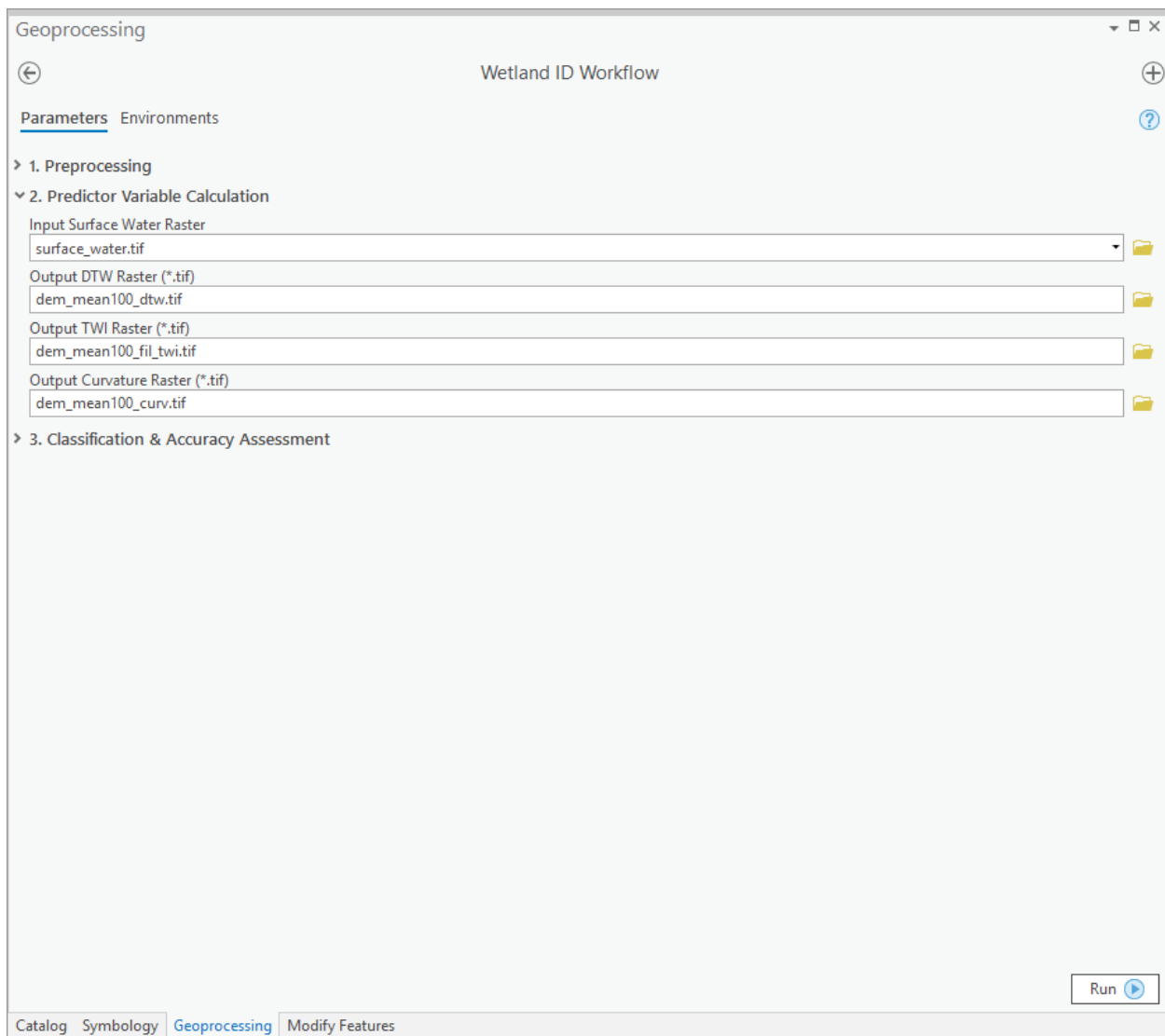
2. Predictor Variable Calculation

3. Classification & Accuracy Assessment

Run

Catalog Symbology Geoprocessing Modify Features

Since we are using a mean smoothing method, the smoothing iterations parameter can be disregarded



Geoprocessing

Wetland ID Workflow

Parameters Environments

> 1. Preprocessing

> 2. Predictor Variable Calculation

▼ 3. Classification & Accuracy Assessment

Input Ground Truth Raster (\*.tif)  
ground\_truth.tif

Training Sampling Area Constraint  
train\_constraint.tif

Class Values (comma separated list)  
0, 1

Class Sampling Percentages (comma separated list)  
10, 60

Output Training Raster (\*.tif)  
train\_demo.tif

Output Testing Raster (\*.tif)  
test\_demo.tif

Output Trained Model (\*.JOBLIB)  
D:\Data\WIM\_documentation\MyProject\Layers\model\train\_demo\_model.JOBLIB

Prepared Predictor Variable Raster (\*.tif)  
composite\_demo.tif

Output Variable Importance (\*.txt)  
D:\Data\WIM\_documentation\MyProject\Layers\model\train\_demo\_varimp.txt

Number of Trees  
100

Maximum Number of Features  
auto

Maximum Tree Depth  
None

Class Weights

Target Class	Class Weight
0	1
1	50

Output Prediction Raster (\*.tif)  
train\_demo\_model\_predict.tif

☒ Save Probability Rasters

Output Metrics Directory  
train\_demo\_model\_predict\_metrics

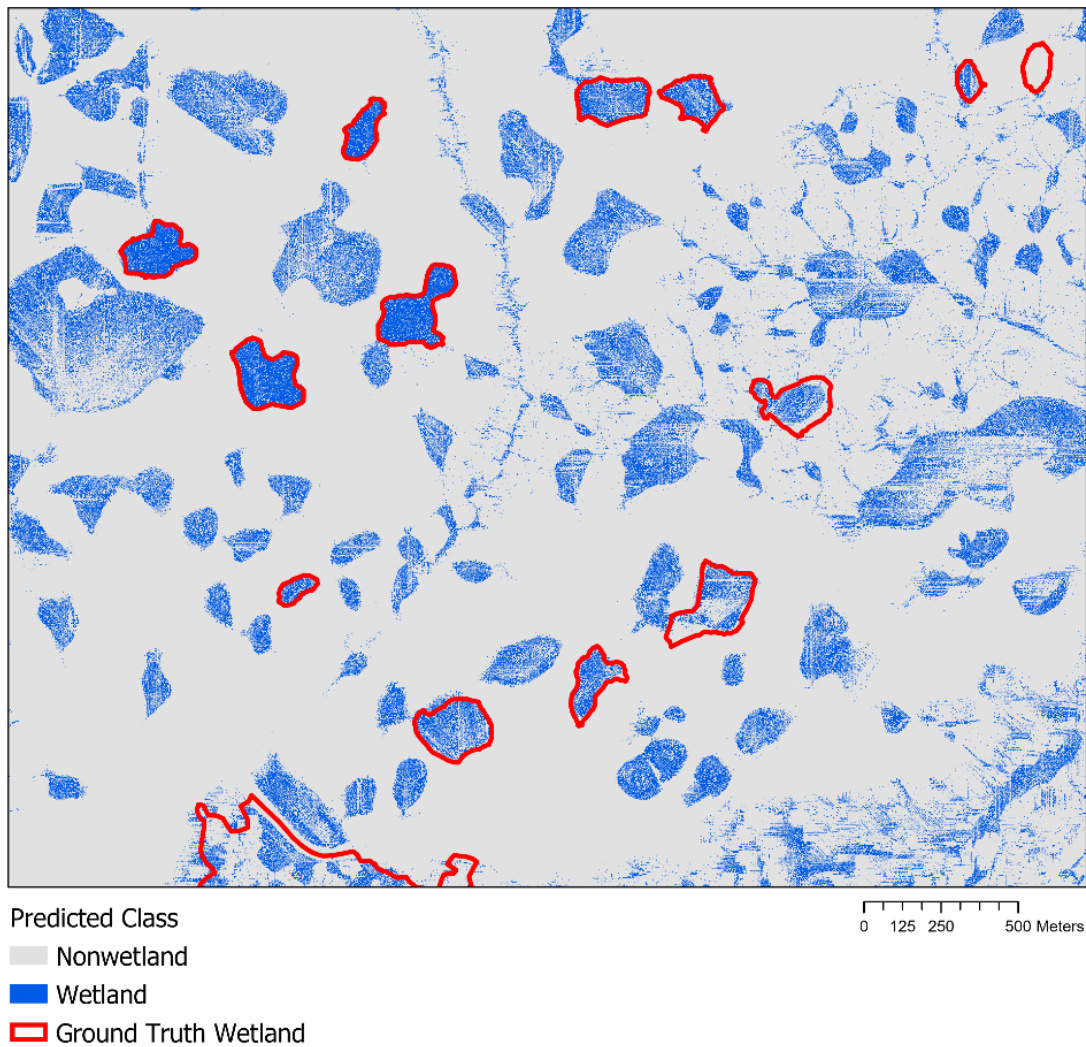
Run

Symbology Modify Features Catalog Geoprocessing

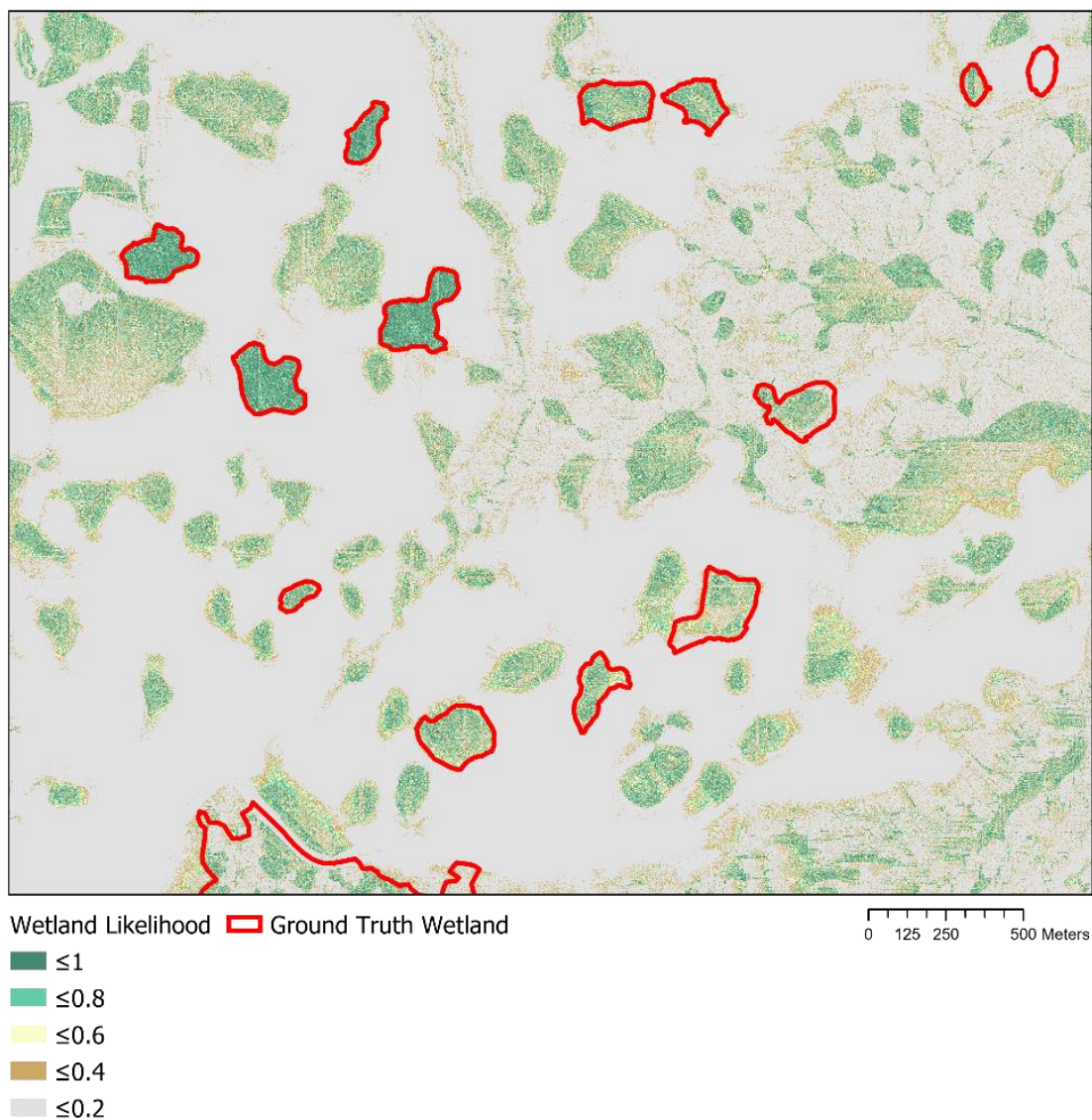
We will have to manually re-run the Assess Accuracy tool outside of this model to include the output probability rasters in the accuracy assessment

### 3.3.10.2 Example output structure

The output data structures mirror those described in previous sections. Below are images of model output created using the parameters outlined here. First, the model prediction, followed by the wetland class probability raster.



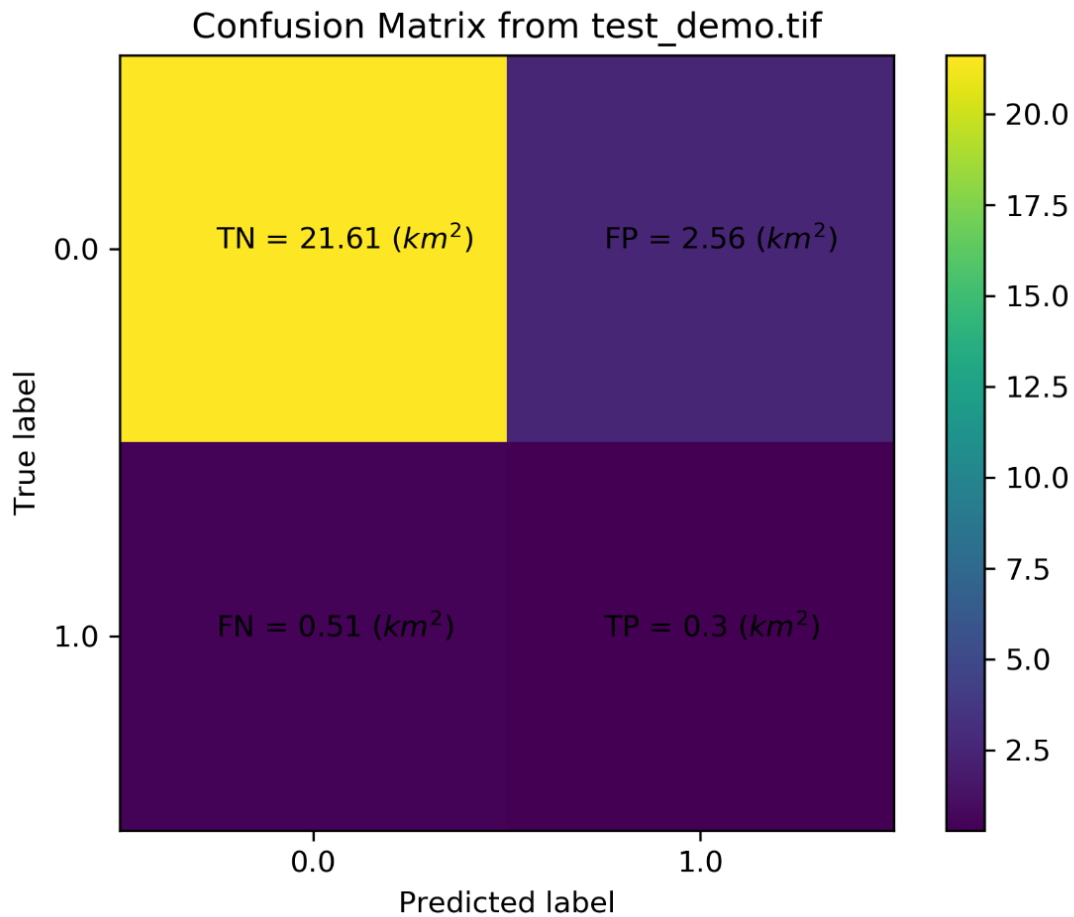
**Figure 10.** An example of wetland and nonwetland predictions produced by the above workflow, with the outline of the ground truth wetlands shown for reference.



**Figure 11.** An example of the wetland likelihood estimated by the above workflow, with the outline of the ground truth wetlands shown for reference.



In addition, the outputs from the accuracy assessment are as follows.



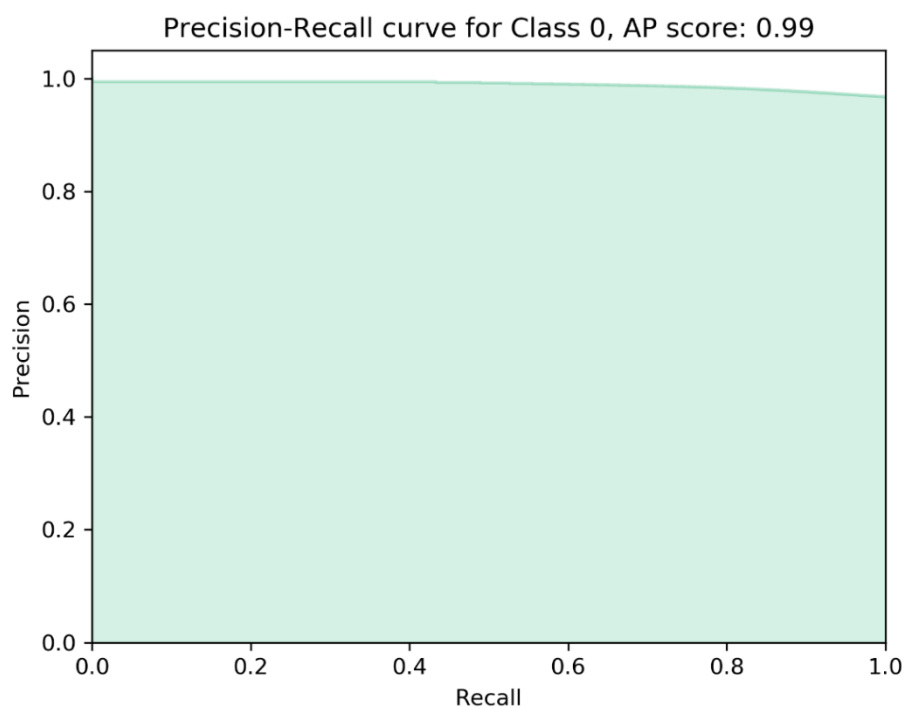
**Figure 12.** The confusion matrix generated by the accuracy assessment for the demo wetland identification application.

summary\_stats.txt - Notepad

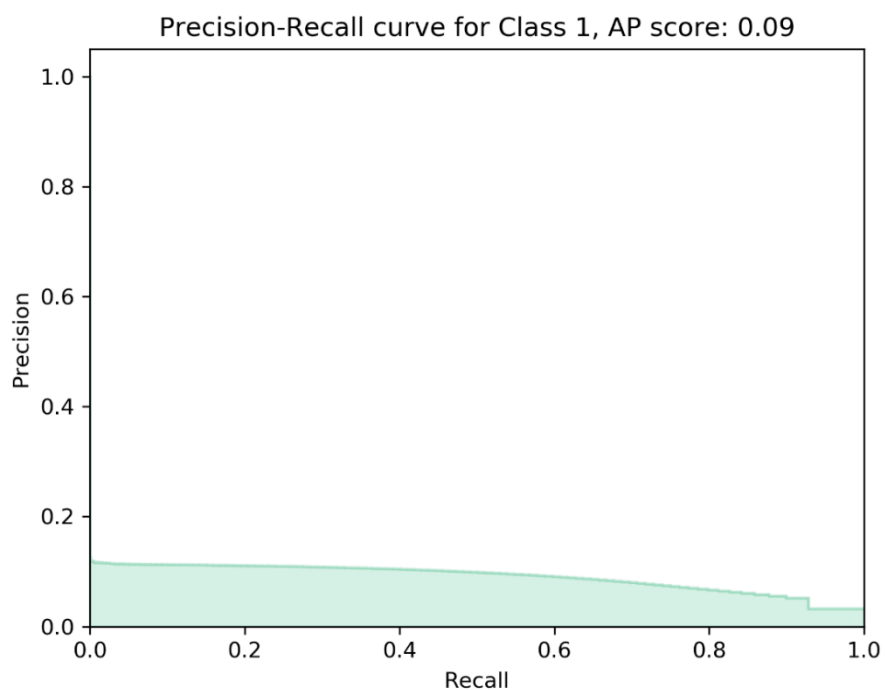
File	Edit	Format	View	Help
	precision	recall	f1-score	support
0.0	0.977	0.894	0.934	41620426
1.0	0.106	0.374	0.165	1390165
accuracy			0.877	43010591
macro avg	0.541	0.634	0.549	43010591
weighted avg	0.949	0.877	0.909	43010591

Windows (CRLF) Ln 1, Col 1 100%

**Figure 13.** The summary statistics report generated by the accuracy assessment for the demo wetland identification application.



**Figure 14. The precision-recall curve for the 0-class predictions (nonwetland class) generated by the accuracy assessment for the demo wetland identification application.**



**Figure 15. The precision-recall curve for the 1-class predictions (wetland class) generated by the accuracy assessment for the demo wetland identification application.**

### 3.4 Results interpretation

Developing one's own wetland model that is best suited to their target landscape and end use will require several iterations of the WIM and adjusting parameters based on the interpretation of model results. It is also recommended that all the provided accuracy metrics are considered collectively when evaluating model performance. For example, keeping in mind the [accuracy disclaimer](#) for the demo application, the demo results can be interpreted as follows.

Application 1 (Section 2.3.9):

- Overall, Model Application 1 was a poor estimator of wetlands.
- The confusion matrix (Figure 6) shows a very high True Negative area and very low True Positive area. This translates to very complete coverage of nonwetland areas and very sparse coverage of wetland areas. While this may lead one to interpret that the model is an accurate estimator of nonwetlands, the significantly greater distribution of nonwetland area than wetland area (i.e., imbalanced land cover scenario) means that an unskilled model that predicted the entire area to be nonwetland would have performed nearly as well.
- These observations are supported by the summary statistics (Figure 7), which are calculated using values from the confusion matrix. Specifically, the precision and recall for class 0 (nonwetland) are suspiciously high. This again points to the ease of correctly identifying nonwetland areas because they are so abundant in the landscape. Moreover, the very low recall and precision for class 1 (wetland) summarize that few wetlands were detected and few of the wetland predictions made were correct.
- The precision-recall curves (Figure 8 and Figure 9) also support the model assessments above. Focusing on the curve for class 1, the Average Precision (AP) score is very low. Ideally, the AP score would be closer to 1, representing a model that sacrifices little precision (i.e., does not make significantly more erroneous wetland predictions) as the predictive threshold becomes less strict and more wetland predictions are made. Users could gain further insight from these curves by plotting the baseline performance of a random classifier, which is the horizontal line plotted at the proportion of ground truth occurrences in the landscape (i.e.,  $\frac{\text{[ground truth wetlands]}}{\text{[ground truth wetlands] + [ground truth nonwetlands]}}$ ).

#### Application 2 (Section 2.3.10):

- The second demo application was still a poor identifier of wetlands, but an improvement on Application 1.
- First, we can see that True Positive predictions increased (Figure 12). This is reflected by a higher wetland recall without sacrificing much of the wetland precision score (Figure 13).
- The model for application 2 was still prone to overpredicting nonwetland area, exemplified by suspiciously high nonwetland metrics, including the AP score (Figure 14). However, there was a slight improvement to the wetland class AP score (Figure 15), showing that this model became better able to identify wetlands.

Based on the assessments for applications 1 and 2, next steps for model improvement may include:

- Consider alternative training sampling schemes. Although Application 1 used a greater quantity of training data overall, Application 2 outperformed with less training data overall but a lower sampling of nonwetland area. Literature supports undersampling the majority class in an imbalanced scenario can improve the detection rate of the minority class.
- Remove the training constraint or change its location. It is possible that the wetlands found within the training constraints shared between the applications are not representative of the types of wetlands in the testing area. By sampling more types of wetlands, the model can learn a more robust set of wetland characteristics.
- Try additional trials of the DEM preprocessing phase. Literature shows that the best performing smoothing method and scale for the DEM may vary between the topographic predictor variables. Also, depending on the size of wetlands typical to the target area, predictor variables may model the wetland hydrology better when derived at coarser or finer scales (i.e., with less or more microtopographic features included)
- Overall, improve the quality of the ground truth datasets. In both applications, we were limited by estimating the extents of the wetland delineation. For this reason, it is possible that additional wetlands exist in the ground truth dataset. This could lead to unknowingly training the model with false ground truth labels. Improving the quality of this data would not only improve the model training, but also provide a more representative accuracy assessment for the same reasons.

## 4.0 Individual tool help

### 4.1 Smooth High-Resolution DEM

This tool creates a smoothed DEM using one of four methods: mean, median, Gaussian, or Perona-Malik. These four methods were selected for their common use in hydrology-related applications. Smoothing is used to blur DEMs to remove the changes in elevation that are too small to indicate features of interest (i.e., microtopographic noise), which are ubiquitous in high-resolution DEMs. The scale of smoothing, or proxy for the scale of smoothing, determines the size of features that are preserved. Users should choose this parameter based on the scale of the hydrologic parameter that is being extracted.

#### 4.1.1 Input High-Resolution DEM

DEM raster used for definition of the terrain surface. The DEM must be in TIFF format. Vertical units should be in meters, and horizontal units will be converted to meters if necessary. This input DEM is also used to define the cell size and extent for all subsequent raster creation.

#### 4.1.2 Smoothing Method

The smoothing method options include: “mean”, “median”, “Gaussian”, and “Perona-Malik.” These can be chosen from the pull-down, or entered as a string outside of the tool user interface. Users should carefully consider which of these smoothing methods are most appropriate for their application.

#### 4.1.3 Output Smoothed DEM

The output name for the smoothed DEM. This is auto filled based on the smoothing method and smoothing rate chosen and saved to the “secondary\_outputs” folder. The DEM must be in TIFF format.

#### 4.1.4 Rate of Smoothing: Smoothing Width or Perona-Malik Iterations (Optional)

There are two possible parameters to control the rate of smoothing: Smoothing Width or Perona-Malik Iterations.

- a. If users choose mean, median, or Gaussian smoothing methods, then the smoothing width parameter will be enabled. If left blank, the default value is 10 m, however users are encouraged to evaluate the benefit of other values, as they can significantly affect the accuracy of the WIM.

- b. As the name suggests, the Perona-Malik Iterations parameter becomes enabled when users select the Perona-Malik smoothing method. If left blank, the default value is 50 iterations, however users are encouraged to evaluate the benefit of other values, as they can significantly affect the accuracy of the WIM.

## **4.2 Hydrocondition High-Resolution DEM**

This tool resolves topographic depressions using Fill, a widely used method for extracting hydrologic parameters from DEMs. Topographic depressions may represent erroneous or actual features. In the current implementation, the only available method is Fill. However, future implementations may include methods that are better suited to hydrologically correcting high-resolution DEMs. Inputs and outputs are shown in the example run.

### **4.2.1 Input High-Resolution DEM**

DEM raster used for definition of the terrain surface. The DEM must be in TIFF format. It is intended that this will be the smoothed DEM.

### **4.2.2 Conditioning Method**

The conditioning method options include “Fill” and “A\* Least-Cost Path”; however, only Fill is available in the current implementation.

### **4.2.3 Output Conditioned DEM**

The output name for the conditioned DEM. This is auto filled based on the hydroconditioning method chosen. The default name retains the root name of the input DEM and is saved to the “secondary\_outputs” folder. The DEM must be in TIFF format.

## 4.3 Calculate Depth to Water Index

This tool calculates the cartographic depth-to-water index (DTW) to be subsequently used as a predictor of wetland areas. The DTW, developed by Murphy et al. (2007), is a soil moisture index based on the assumption that soils closer to surface water, in terms of distance and elevation, are more likely to be saturated.

### 4.3.1 Input Smoothed DEM

DEM raster used for definition of the terrain surface. The DEM must be in TIFF format. It is intended that this will be the smoothed DEM. This DEM is used to calculate DTW components if they are not passed as optional parameters. If a pre-calculated slope raster is provided, the input DEM is not used in the code, other than to autofill the output DTW raster name.

### 4.3.2 Output DTW Raster

The output name for the DTW raster. This is auto filled to retain the DEM preprocessing steps by including the base name of the input DEM. By default, this is saved to the “predictor\_variables” folder. The DTW must be in TIFF format.

### 4.3.3 Save Intermediate Outputs

Boolean option to save DTW components derived from the input DEM. If this option is checked AND a pre-calculated slope raster is NOT given, a DTW slope raster will be saved to “secondary\_outputs” with the suffix “\_dtwSLP.tif” and the DEM base name. Future implementations will have similar functionality for a DEM-derived surface water raster.

### 4.3.4 Output DTW Slope Raster

The output name for the DTW Slope raster. This is auto filled to retain the DEM preprocessing steps by including the base name of the input DEM. By default, this is saved to the “secondary\_outputs” folder.

### 4.3.5 Input DTW Slope Raster (Optional)

Slope raster needed for the DTW calculations. The DTW slope raster is in units of m/m (rise/run) and has a very small constant (0.001) added to all values to avoid confusion between a zero-slope area and the source locations (recognized by values equal to zero internally). If the user chooses to use a pre-calculated slope raster, they must ensure it meets the specifications above, or the output may be inaccurate.



#### 4.3.6 Input Surface Water Raster (Optional)

Surface water raster needed for DTW calculation. Cells that represent surface water must have data, while all other cells are null. This raster does not need to be in TIFF format. **Note, in this current implementation, the Input Surface Water Raster is required.**

## **4.4 Calculate Curvature**

This tool calculates the curvature (i.e., the second derivative) of the land surface to be subsequently used as a predictor of wetland areas. Curvature can be used to describe the degree of convergence and acceleration of flow (Moore et al., 1991). Curvature calculation for the WIM requires one input: a high-resolution DEM.

### **4.4.1 Input Smoothed DEM**

DEM raster used for definition of the terrain surface. It is intended that this will be the smoothed DEM. This DEM is used to calculate the curvature.

### **4.4.2 Output Curvature Raster**

The output name for the Curvature raster. This is auto-filled to retain the DEM preprocessing steps by including the base name of the input DEM. By default, this is saved to the “predictor\_variables” folder. The raster must be in TIFF format.

## **4.5 Calculate Topographic Wetness Index**

This tool calculates the topographic wetness index (TWI) to be subsequently used as a predictor of wetland areas. The TWI relates the tendency of an area to receive water to its tendency to drain water.

### **4.5.1 Input Hydroconditioned DEM**

Hydrologically corrected DEM raster used for definition of the terrain surface. The DEM must be in TIFF format. It is intended that this DEM will also have been smoothed during preprocessing. This DEM is used to calculate TWI components if they are not passed as optional parameters. If these components are provided, the input DEM is not used in the code, other than to autofill the output TWI raster name.

### **4.5.2 Output TWI Raster**

The output name for the TWI raster. This is auto filled to retain the DEM preprocessing steps by including the base name of the input DEM. By default, this is saved to the “predictor\_variables” folder. The TWI must be in TIFF format.

### **4.5.3 Save Intermediate Outputs**

Boolean option to save TWI components derived from the input DEM. If this option is checked AND pre-calculated TWI inputs are NOT given, a TWI slope raster and TWI specific catchment area raster will be saved to “secondary\_outputs” with the suffixes “\_twiSLP.tif” and “\_twiSCA.tif”, respectively. The DEM base name will precede these.

### **4.5.4 Output TWI Slope**

The output name for the TWI Slope raster. This is auto filled to retain the DEM preprocessing steps by including the base name of the input DEM. By default, this is saved to the “secondary\_outputs” folder.

### **4.5.5 Output TWI Specific Catchment Area**

The output name for the TWI Specific Catchment Area raster. This is auto filled to retain the DEM preprocessing steps by including the base name of the input DEM. By default, this is saved to the “secondary\_outputs” folder.

#### 4.5.6 Input TWI Slope Raster (Optional)

Slope raster needed for the TWI calculations. The TWI slope raster has units of  $\frac{[L]}{[L]} \left( \frac{[rise]}{[run]} \right)$  and has a very small constant (0.001) added to all values. If zero values are present, the TWI raster will have undefined values. The TWI Slope must be in TIFF format. If the user chooses to use a pre-calculated slope raster, they must ensure it meets the specifications above, or the output may be inaccurate.

#### 4.5.7 Input TWI Specific Catchment Area Raster (Optional)

Specific Catchment Area raster needed for the TWI calculations. The specific catchment area raster has units of  $\frac{[L^2]}{[L]}$  and is calculated as the contributing area (flow accumulation raster \* area per cell) divided by the cell width (i.e., cell size). Flow accumulation is calculated using the D-Infinity algorithm (Tarboton, 1997). In addition, a constant of 1 is added to the flow accumulation raster, ensuring that each cell receives flow from at least itself and avoiding undefined values in the TWI raster. The TWI Specific Catchment Area must be in TIFF format. If the user chooses to use a pre-calculated slope raster, they must ensure it meets the specifications above, or the output may be inaccurate.

## **4.6 Train Test Split**

This tool splits the input verification raster into two subsets: training and testing data. The training raster is created by randomly sampling a user-defined percentage of each ground truth class. The testing raster is the complement of the training raster, comprised of the remaining, unsampled cells.

### **4.6.1 Input Ground Truth Data**

The ground truth raster containing the true locations of each target class, in this case wetland and nonwetland. Each class must be represented by a unique integer value, beginning with value 0 and increasing for each class. In this demo, the ground truth raster contains values of 0 (nonwetland area) and 1 (wetland area).

### **4.6.2 Class Values**

A comma separated list of the values that correspond to each class in the ground truth raster.

### **4.6.3 Percent to Sample from Each Class**

A comma separated list of the percent of each class to use as training data. A random selection of the number of cells corresponding to each percentage will be selected for training. The index of each percent value must match the index of the class value it is intended for.

### **4.6.4 Output Training Raster**

The output name for the training raster. By default, this is saved to the “model” folder as “train.tif”. This raster must be in TIFF format.

### **4.6.5 Output Testing Raster**

The output name for the testing raster. By default, this is saved to the “model” folder as “test.tif”. This raster must be in TIFF format.

### **4.6.6 Training Sampling Area Constraint (Optional)**

This raster should be of Boolean type (integer, 1’s and 0’s), where the True values indicate the areas where the tool should sample training data from. This raster must match the extents of the ground truth raster.

## **4.7 Train Random Trees**

This tool executes the training phase of the Random Trees algorithm. In this phase, the algorithm takes bootstrap samples of the training dataset, including the labeled cells in the training raster and the predictor variables. A decision tree is created from each bootstrap sample, and all are used to learn the indicators of the ground truth classes based on information from the predictor variables.

### **4.7.1 Input Training Raster**

Raster containing the ground truth labels for cells. This should be a subset of the ground truth data. This raster must be in TIFF format.

### **4.7.2 Input Predictor Variables Raster**

One or more rasters containing the predictor variable information, which are intended to differentiate between ground truth classes.

### **4.7.3 Output Trained Model**

The output name for the trained Random Trees model. This is auto filled to retain the base name of the training raster used to create the model. By default, this is saved to the “model” folder. The trained model must be in JOBLIB format.

### **4.7.4 Prepared Predictor Variable Raster**

The output name for the prepared predictor variable raster. If more than one raster is passed to the Input Predictor Variables Raster parameter, this output name corresponds to a composite of all predictor variables. If only one predictor variable is passed, this output name corresponds to a copy of the input predictor variable, and this raster can be disregarded. This is auto-filled as “composite.tif” and saved to the “predictor\_variables” folder.

### **4.7.5 Output Variable Importance**

The output name for the variable importance file. This is auto filled to retain the base name of the training raster used to create the model. By default, this is saved to the “model” folder. This file must be in TXT format.

#### **4.7.6 Number of Trees (Optional)**

Input number of decision trees to grow in the Random Trees model. If this parameter is left blank or unaltered, a default value of 100 is used.

#### **4.7.7 Maximum Tree Depth (Optional)**

The maximum depth of each tree in the Random Trees model. If this parameter is left blank or unaltered, a default value of “None” will be used. The value of None will result in decision tree nodes being expanded until all leaves are pure.

#### **4.7.8 Maximum Number of Features (Optional)**

The maximum number of features that are considered when a decision tree node is split. If this parameter is left blank or unaltered, a default value of “auto” will be used. The value of “auto” will set the max features number to be the square root of the number of features.

#### **4.7.9 Class Weights (Optional)**

The weight to assign to each class when calculating the penalty for misclassifying that class during training. This parameter is entered as a value table where column 0 is the class value and column 1 is the weight for that class. If left blank or unaltered, a default value of “balanced” is used. The value “balanced” automatically adjusts class weights to be inversely proportional to class frequencies in the training dataset.

## **4.8 Run Random Trees**

This tool uses the trained Random Trees model to predict the class for the cells of the input predictor variables. Predictions do not need to be made in the same area for which the model was trained, but the calculation of the predictor variable(s) must be the same. In this case, a model can be trained using the DTW, Curvature, and TWI for one area and used to make predictions for a new area, if DTW, Curvature, and TWI have also been derived and are used as inputs

### **4.8.1 Input Predictor Variables Raster**

Raster containing the predictor variables that will be used by the model to differentiate between classes. These predictor variables must match those that were used to train the model. If a composite of input predictor variables was created from the Train Random Trees tool, the output Prepared Predictor Variables Raster must be used here. If only one predictor variable was used to train the model, that same raster must be used here. This raster must be in TIFF format.

### **4.8.2 Input Trained Model**

The trained model resulting from running the Train Random Trees tool. This model was trained using the same set of predictor variables present in the Input Predictor Variables Raster. The trained model must be in JOBLIB format.

### **4.8.3 Output Prediction Raster**

The output name for the prediction raster, where each cell has been assigned a predicted class based on the input predictor variables raster. This is auto filled to retain the base name of the trained model. By default, this is saved to the “outputs” folder. This raster must be in TIFF format.

### **4.8.4 Save Probability Rasters**

Boolean option to save probability rasters. If True, a raster is created for each target class and saved to the “outputs folder.” The rasters are saved with the pattern: “[trained model base name]\_proba[class value].tif”.



## **4.9 Assess Accuracy**

Given the predicted output from the preceding step and a corresponding raster with ground truth labels for the cells (i.e., the testing raster), this tool generates accuracy metrics that summarize the model's ability to predict wetlands and nonwetlands. Additional metrics are generated if the user includes class probability rasters. Accuracy metrics were chosen to avoid misleading assessments of imbalanced ground truth classes, which is typical of wetland/nonwetland distributions.

### **4.9.1 Input Testing Raster**

Raster containing the ground truth labels for cells. This should be a subset of the ground truth data that was NOT used for training, intended to be the testing raster output from the Train Test Split tool. The location of the non-null cells of this raster dictates where the accuracy metrics are calculated for. This raster must be in TIFF format.

### **4.9.2 Input Prediction Raster**

Raster containing the class predictions generated by the Random Trees model, i.e., the output of Run Random Trees. If this raster does not have the same extents and number of cells as the input testing raster, the tool will fail. Ensure that both rasters were created from the same origin rasters and in the same projections. This raster must be in TIFF format.

### **4.9.3 Output Metrics Directory**

The output name for the directory where accuracy metrics will be stored. This is auto filled to retain the base name of the trained model and located at the same directory level as "Layers".

### **4.9.4 Optional Accuracy Outputs (Optional)**

An optional value table to enter the file path to probability rasters (column 1) and the class value that they correspond to (column 0). If inputs are passed, additional precision-recall curves will be calculated for these rasters and saved to the output metrics directory. By default, the precision-recall curve(s) will be saved to PNG format with the pattern: "prec\_rec\_curve\_[class value]."

## 5.0 References

- Beven, K. J., & Kirkby, M. J. (1979). A physically based, variable contributing area model of basin hydrology/Un modèle à base physique de zone d'appel variable de l'hydrologie du bassin versant. *Hydrological Sciences Journal*, 24(1), 43-69.  
<https://doi.org/10.1080/02626667909491834>
- Grimaldi, S., Nardi, F., Di Benedetto, F., Istanbuluoglu, E., & Bras, R. L. (2007). A physically-based method for removing pits in digital elevation models. *Advances in Water Resources*, 30(10), 2151-2158. <https://doi.org/10.1016/j.advwatres.2006.11.016>
- Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2), 100-107. [10.1109/TSSC.1968.300136](https://doi.org/10.1109/TSSC.1968.300136)
- Jenson, S. K., & Domingue, J. O. (1988). Extracting topographic structure from digital elevation data for geographic information system analysis. *Photogrammetric engineering and remote sensing*, 54(11), 1593-1600.
- Lashermes, B., Foufoula-Georgiou, E., & Dietrich, W. E. (2007). Channel network extraction from high resolution topography using wavelets. *Geophysical Research Letters*, 34(23). <https://doi.org/10.1029/2007GL031140>
- Lindsay, J. B. (2016). Efficient hybrid breaching-filling sink removal methods for flow path enforcement in digital elevation models. *Hydrological Processes*, 30(6), 846-857.  
<https://doi.org/10.1002/hyp.10648>
- Lindsay, J. B., & Creed, I. F. (2005). Removal of artifact depressions from digital elevation models: towards a minimum impact approach. *Hydrological Processes: An International Journal*, 19(16), 3113-3126. <https://doi.org/10.1002/hyp.5835>
- Metz, M., Mitsova, H., & Harmon, R. S. (2011). Efficient extraction of drainage networks from massive, radar-based elevation models with least cost path search. ULM UNIV (GERMANY FR). <https://doi.org/10.5194/hess-15-667-2011>
- Moore, I. D., Grayson, R. B., & Ladson, A. R. (1991). Digital terrain modelling: a review of hydrological, geomorphological, and biological applications. *Hydrological processes*, 5(1), 3-30. <https://doi.org/10.1002/hyp.3360050103>
- Murphy, P. N., Ogilvie, J., Connor, K., & Arp, P. A. (2007). Mapping wetlands: a comparison of two different approaches for New Brunswick, Canada. *Wetlands*, 27(4), 846-854.  
[https://doi.org/10.1672/0277-5212\(2007\)27\[846:MWACOT\]2.0.CO;2](https://doi.org/10.1672/0277-5212(2007)27[846:MWACOT]2.0.CO;2)

- Planchon, O., & Darboux, F. (2002). A fast, simple and versatile algorithm to fill the depressions of digital elevation models. *Catena*, 46(2-3), 159-176. [https://doi.org/10.1016/S0341-8162\(01\)00164-3](https://doi.org/10.1016/S0341-8162(01)00164-3)
- O'Callaghan, J. F., & Mark, D. M. (1984). The extraction of drainage networks from digital elevation data. *Computer vision, graphics, and image processing*, 28(3), 323-344. [https://doi.org/10.1016/S0734-189X\(84\)80011-0](https://doi.org/10.1016/S0734-189X(84)80011-0)
- O'Neil, G. L., Goodall, J. L., & Watson, L. T. (2018). Evaluating the potential for site-specific modification of LiDAR DEM derivatives to improve environmental planning-scale wetland identification using Random Forest classification. *Journal of hydrology*, 559, 192-208. <https://doi.org/10.1016/j.jhydrol.2018.02.009>
- O'Neil, G. L., Saby, L., Band, L. E., & Goodall, J. L. (2019). Effects of LiDAR DEM smoothing and conditioning techniques on a topography-based wetland identification model. *Water Resources Research*, 55, 4343– 4363. <https://doi.org/10.1029/2019WR024784>.
- Passalacqua, P., Do Trung, T., Foufoula-Georgiou, E., Sapiro, G., & Dietrich, W. E. (2010). A geometric framework for channel network extraction from lidar: Nonlinear diffusion and geodesic paths. *Journal of Geophysical Research*, 115, F01002. <https://doi.org/10.1029/2009JF001254>.
- Sangireddy, H., Stark, C. P., Kladzyk, A., & Passalacqua, P. (2016). GeoNet: An open source software for the automatic and objective extraction of channel heads, channel network, and channel morphology from high resolution topography data. *Environmental Modelling & Software*, 83, 58-73. <https://doi.org/10.1016/j.envsoft.2016.04.026>.
- Scikit-learn Developers. (2017a). Ensemble Methods. Retrieved August, 2018 from <http://scikit-learn.org/stable/modules/ensemble.html#forest>
- Scikit-learn Developers. (2017b). Model evaluation: quantifying the quality of predictions. Retrieved August 2018, from [http://scikit-learn.org/stable/modules/model\\_evaluation.html](http://scikit-learn.org/stable/modules/model_evaluation.html)
- Wang, L., & Liu, H. (2006). An efficient method for identifying and filling surface depressions in digital elevation models for hydrologic analysis and modelling. *International Journal of Geographical Information Science*, 20(2), 193-213. <https://doi.org/10.1080/13658810500433453>
- Woodrow, K., Lindsay, J. B., & Berg, A. A. (2016). Evaluating DEM conditioning techniques, elevation source data, and grid resolution for field-scale hydrological parameter extraction. *Journal of hydrology*, 540, 1022-1029. <https://doi.org/10.1016/j.jhydrol.2016.07.018>