Ology Model Trainer

A field guide to training archaeological site-sensitivity models from your own LiDAR — how the model works, what it does, and how to use the desktop trainer.

This guide is written for archaeologists and cultural-resource professionals. It assumes no machine-learning background. It explains, in plain language, what the trainer is doing under the hood, why the method is sound, how to run it, and how to read the results — including the held-out validation map that demonstrates the model actually works.

> **What you produce**
> A single model file (.json) that imports into the Ology iPhone/iPad app and predicts site sensitivity across any area — fully offline, no cell signal, no cloud. Train once on ground you know; predict anywhere.

Contents

1 What the model does

The trainer builds a predictive site-sensitivity model. It learns, from the places where sites are already known, what the terrain tends to look like where people lived — then it scores every patch of a new area for how much that ground resembles the known-site terrain. The output is a heat surface: brighter means more likely to reward survey.

This is fundamentally a question about landscape position, not about detecting objects. The model never tries to “see” a house pit or a scatter in the imagery. Instead it asks: is this spot the kind of place — near reachable water, on a flood-safe bench, gently sloped, sheltered — that people historically chose? That is a question terrain can answer, which is why it works even for sites that leave no visible trace on the surface.

> **Why not “AI that finds sites in the lidar”?**
> Deep-learning image detection (the technique behind headlines about spotting Maya structures) only works when a site has a distinctive shape in the terrain — a mound, a platform, an earthwork. Most sites in a place like Mendocino — lithic scatters, many middens — have no such shape. For predicting where sites are, rather than detecting a visible form, the current research is clear and consistent: gradient-boosted decision trees (this trainer’s default) match or beat neural networks on this kind of terrain data. This trainer uses the tool that actually wins.

A note on what the model is not: it is a survey-prioritisation aid, not a site locator. A high score means “worth looking here first,” not “a site is here.” Site locations remain confidential and the model output is a probability surface, not a confirmed record.

2 The predictors — what the model looks at

Every predictor is derived from your LiDAR DEM (and, where available, hydrography, vegetation, and soils). Each captures a piece of settlement logic that archaeologists already reason about:

Slope steepness	Habitable flats vs. unusable steep ground.
Distance to water	Proximity to streams and drainages (NHD + DEM-derived).
Distance to confluence	Stream junctions — often favoured living spots.
Cost-distance to water	Effort to reach water (uphill counts more than flat).
Height above water	Flood-safe terraces above the nearest drainage.
Distance to predicted spring	Nearness to likely seeps/springs found in the DEM (optional — see §5).
Landform position (MSTP)	Ridges, benches, terraces — multi-scale topography.
Local prominence	Knolls and benches standing above their surroundings.
Human-scale flatness	Is there a tent-sized usable patch, even on broken ground?
Terrain ruggedness	Smooth benches vs. broken, dissected ground.
Solar aspect (southness)	South-facing warmth — winter sun.
Shelteredness	Tucked below the ridgeline, out of exposure.

Vegetation (optional)	Detailed CALVEG type — alliance-level classes, not just broad conifer/hardwood/shrub (see §5).
Soils (optional)	USDA SSURGO soil map units, by name (see §5).

The terrain predictors always come from your DEM — nothing to download. Predicted springs are also DEM-derived, so they work offline in the app too. Vegetation and soils are fetched from public services when you enable them; because a phone can’t rebuild those two offline, models that use them list them as “Missing here” in the app and apply every other factor. The model still works.

3 How the model learns

The trainer offers two model types. You can train both and compare.

Frequency Ratio (FR) — the explainable baseline

FR is transparent arithmetic. For each predictor it bins the landscape (e.g. bands of slope) and asks: what fraction of known sites fall in this band, versus what fraction of the whole landscape is this band? If sites are over-represented, that band gets a ratio above 1. The scores across predictors combine into a sensitivity surface. Every number is auditable — you can say exactly why a spot scored high ("sites are 3.4× more likely on flood-safe terraces"). It is fast and the right first model.

Gradient-Boosted Trees (GBT) — the default

GBT learns interactions that FR cannot: "south-facing matters, but only on gentle slopes near water." It builds hundreds of small decision trees, each correcting the last, into one strong predictor. On tabular terrain data like this, gradient boosting is the current state of the art — it consistently outperforms deep neural networks in independent benchmarks. It is slightly less explainable than FR, but the trainer still reports each factor's importance so you see what drove the model.

> **Background sampling (the honest-model detail)**
> You have presence data (known sites) but no certified absence. The trainer contrasts sites against a "background" sample of the landscape, drawn stratified across terrain types so the comparison represents the whole area rather than over-sampling common ground. Getting this right matters as much as the choice of algorithm.

4 Making it robust

Three optional features make a GBT model more trustworthy for real survey decisions. All are toggles in the trainer.

Auto-tune

Instead of fixed settings, the trainer searches a small set of model configurations and keeps the one that scores best on spatial held-out tests — that is, the settings that best predict ground the model didn't train on, not the settings that best memorise. Adds a few minutes.

Ensemble

Trains several models (five by default) on different random draws and averages them. Averaging reduces the luck-of-the-draw variance in any single model, giving a steadier surface. The ensemble exports to the same file format, so the app runs it with no changes.

Uncertainty

Because an ensemble is several models, their disagreement at each spot is a genuine confidence measure. "High sensitivity and the models agree" is a stronger survey target than "high but the models disagree." The trainer reports the mean uncertainty across the area and stores it with the model.

5 Water, springs, vegetation & soils

Four data layers extend the terrain model. The first two are DEM-derived and work offline in the app; the last two are downloaded during training.

Water (always on)

The National Hydrography Dataset (mapped streams and springs) is merged with drainages traced from your DEM (depression-fill + D8 flow accumulation). This is the backbone of distance-to-water, height-above-water, and confluence distance, so it can't be switched off.

Predicted springs (optional, DEM-derived)

Not every spring is on a map. This option finds likely spring and seep locations from the DEM itself, using a standard geomorphic proxy: channel heads in concave valley heads — the upstream ends of the drainage network, in hollows where hillslope flow first concentrates and the water table is most likely to reach the surface. Detected springs join the water network and add a "distance to predicted spring" factor.

> **Honest by design — it turns itself off if it doesn't work**
>
> A predicted spring is a good place to *check*, not a guaranteed spring. So the trainer validates its own output: channel heads should be a small fraction of the landscape. If your DEM yields none (too flat, or too small an extent) or an implausible flood of them (a sign of DEM noise or wrong units), springs are disabled automatically and the model trains without them — they never silently pollute the result. The training report states exactly what happened.

Vegetation (optional, CALVEG) — now finer

Vegetation is fetched from the USFS CALVEG existing-vegetation service for California. Earlier builds keyed on the broad cover field (about nine classes — conifer / hardwood / shrub / herb / water…), which is why a large area came back with only a handful of vegetation types. The trainer now prefers the detailed regional-dominance field (alliance-level classes — many dozens over a big extent) and only falls back to the broad field where the detailed one is genuinely absent, in which case the log says so. Very rare classes (a few cells) are pooled into a single "Other vegetation" bucket so their statistics stay stable, while every well-represented class keeps its own resolution — and any class your sites actually sit on is always kept.

Soils (optional, USDA SSURGO)

Soils are fetched from USDA Soil Data Access. The trainer uses the fast, robust query pattern USDA recommends — ask for the intersecting SSURGO map-unit keys, pull a readable name for each (e.g. "Holland loam, 15 to 30% slopes"), then fetch the polygon geometry in chunks — instead of a single heavy spatial query that previously timed out on large extents (and could mix in coarse STATSGO). The result is a named soil-map-unit predictor, with the same rare-class pooling as vegetation. On any failure the run simply continues without soils and the report says why.

App note: current app builds list vegetation and soils under "Missing here" and skip them when predicting on the phone (it can't rebuild those two offline). Desktop training and the held-out validation map still use them, so they sharpen the factor findings and the accuracy read-out you report.

6 Proving it works — the held-out validation map

A single accuracy number is easy to distrust. The validation map is the visual proof. Tick **Held-out validation map** before training and the trainer performs an honest experiment:

- It splits your area into a training region and a spatially separate held-out region — placing the split so a meaningful share of your real sites falls in each.
- It trains on the training region only, so the held-out ground is genuinely unseen.
- It predicts the held-out region and checks: do the real sites there land in the model's high-scoring zones?

You get a map (the held-out sites drawn as circles over the predicted heat surface) and a headline number: of the held-out sites, the model's top-scoring 20% of that ground captured N%. A model with no skill captures about 20%. A good model captures far more — several times better than chance on ground it never saw. That is the figure, and the picture, to show a reviewer or a SHPO.

> **How to read the lift number**
> "Lift 3×" means the top 20% of predicted ground was three times richer in real held-out sites than a random 20% would be. Higher is better. If the held-out number is weak while the whole-area number looks strong, trust the held-out number — it is the honest test of whether the model transfers to new ground.

7 Using the GUI, step by step

The trainer window has three columns: your inputs (left), the live pipeline (centre), and the log / result (right). Work down the four numbered cards on the left, then press Run.

Opening the software

On a Mac, double-click **Ology Model Trainer.app** (or **Ology Predict New Area.app** to go straight to prediction). On Windows, double-click **Ology Model Trainer.bat** or **Ology Predict New Area.bat** — and run **Create Desktop Shortcuts.bat** once to put both on your desktop with the Ology icon. (The plain launch_gui.command / launch_gui_windows.bat scripts do the same thing and are kept for troubleshooting.)

1 · Site data	Choose the layer with your known sites (CSV, KML, KMZ, SHP, GPKG, GeoJSON). Re-run Convert to CSV with the current kit to carry polygon footprints and habitation/lithic routing.
2 · LiDAR DEM	Point at your elevation data — a folder of GeoTIFF tiles, one GeoTIFF, or a VRT. The card reports how much of your sites sit on the DEM and shows a hillshade preview of the imported LiDAR area with your sites as amber dots, so you can see at a glance that everything lines up.
3 · Training data	NHD water is always on. Optionally add vegetation (CALVEG), soils (SSURGO), predicted springs (from your DEM), and the held-out validation map.
4 · Model	Name it (the output filenames preview live). Pick FR or GBT, the validation mode (spatial is the honest one), and cell size — start at 3 m before the full 1 m run.

While it runs

The centre pipeline lights each stage as it works, with its own timer; the terrain stage shows tiles done and an ETA; and a heartbeat at the top proves the run isn't frozen during long steps. Stop is always safe — nothing is corrupted, just run again. When it finishes, the Result tab shows the headline and buttons to open the report,

reveal the model file, and open the validation map.

Predict New Area — the desktop “Use model”

What downloads automatically: elevation streams from USGS 3DEP when you give a lon/lat box instead of a local DEM; vegetation (CALVEG) and soils (SSURGO) are fetched for the new area automatically whenever the model uses them; and the water network is derived from the DEM itself (the same depression-fill + flow-accumulation drainage used in training), with an optional extra-hydrography KML if you supplied one at training. If a download fails, that factor applies as neutral and the prediction report says so. The window also shows a hillshade preview of the new area as soon as you pick its DEM.

The **Predict New Area…** button (in the header, and next to the result buttons after a run) is the computer-side twin of the app’s “Use model”: pick any trained .json and any DEM, set the flag-top percentage, and press Predict. It computes the same predictors over the new ground and scores every cell with the model’s own tables (FR) or trees (GBT) — the desktop ranking matches the phone exactly. You get a heatmap PNG (amber = higher sensitivity, flagged zones outlined), a georeferenced GeoTIFF of the 0–1 score surface for QGIS/ArcGIS, a KML of the flagged zones you can AirDrop to the phone or open in Google Earth, and a prediction report. If the model uses vegetation or soils and they can’t be fetched for the new area, those factors apply as neutral — the same “Missing here” behaviour as the app.

8 Reading the outputs

- **<name>.json** — the model. AirDrop or copy it to the iPhone/iPad, then in Ology: Terrain ■ Archaeological Site Sensitivity ■ Use model ■ import (↓). It predicts fully offline on any area you open.
- **<name>_training_report.txt** — the receipt: inputs, every setting, per-stage timings, the predictor list, the top factor findings in plain English, and the accuracy read-out. It now also records whether springs were kept or auto-disabled, how many CALVEG classes were used, and how many SSURGO map units were fetched.
- **<name>_validation.html** and **_validation_map.png** — the held-out validation report and map (when enabled). The HTML is self-contained; open it in any browser.

In the app, the model produces a sensitivity heat map you can adjust (the “flag top” percentage sets how much of the area becomes high-probability zones), and you can turn the flagged zones into labelled polygon layers to navigate to and export.

On the desktop, **Predict New Area** writes its own set: **<model>_prediction.png** (the heatmap), **.tif** (georeferenced score surface for GIS), **_flagged.kml** (the top-N% zones as polygons), and **_predict_report.txt** (what was applied, what was missing, and the score distribution).

9 Ethics, limits, and honest caveats

- **A prioritisation aid, not a site map.** High score = survey here first, not “a site is here.” Ground-truth everything.
- **Confidentiality.** Site locations are protected under NHPA/ARPA and state law. Model outputs derived from them should be handled with the same care.
- **Garbage in, garbage out.** The model learns from the sites you give it. Biased or sparse training sites produce a biased model. The held-out map helps you see this.
- **Predicted springs are hypotheses.** They flag likely seep locations from terrain; confirm them in the field before treating them as water sources.
- **It predicts terrain preference, not culture.** It cannot know site type, age, or significance — only that ground resembles known-site ground.
- **Transfer has limits.** A model trained in one landscape may not transfer to very different terrain. Retrain for new regions; read the spatial validation number.

10 Troubleshooting & setup

Blank window on macOS	Apple’s built-in Tk is broken. Install Python from python.org (bundles a working Tk); the launcher prefers it automatically.
“Needs rasterio” on the DEM	Click Install requirements (one time). If it fails, your Python may be too new — install Python 3.12/3.13 from python.org and relaunch.
DEM coverage looks low	The bounding-box % can look low from empty corners; the sites-on-DEM % is the one that matters.
Validation map shows 0%	Older builds split by area; the current build splits by sites. Re-run with the current kit.

Only a few vegetation types	Fixed — the trainer now uses the detailed CALVEG field. If a zone truly only exposes the broad cover field, the log says so and vegetation stays coarse there.
Soils didn’t load	The log now says why (nothing in extent, or SDA busy/down). Soils use the robust Soil Data Access pattern; re-run later if the service was down.
Springs turned themselves off	By design: if the DEM yields no channel heads, or far too many, springs are disabled so they can’t pollute the model. Check the DEM’s vertical units and quality.
Runs on Windows?	Yes — install Python 3.12/3.13 from python.org, then double-click launch_gui_windows.bat.

Ology Model Trainer — user guide. Generated for distribution with the training kit. The trainer and app are aids to professional judgement, not replacements for it.