Topic: Classification for Environmental Risk Detection

Course: Machine Learning for Environmental Applications

Learning Outcomes

After completing this unit, students will be able to:

- Apply supervised learning methods to classify environmental risks and hazards.
- Evaluate classification models using precision, recall, F1-score, and AUC metrics.
- Implement and interpret algorithms such as Logistic Regression, SVMs, Random Forests, and CNNs.
- Use model outputs to support environmental monitoring and risk management decisions.

Introduction

Classification models are a central tool in environmental data science, enabling the mapping of diverse observations into discrete risk categories such as contaminated vs. non-contaminated, harmful algal bloom vs. normal conditions, or safe vs. hazardous states. Unlike regression, which predicts continuous outcomes (e.g., pollutant concentrations), classification produces categorical labels that are directly actionable for risk assessment.

Environmental datasets usually integrate multiple sources of information, including:

- Physico-chemical parameters (e.g., pH, dissolved oxygen, nutrient loads),
- Meteorological drivers (e.g., temperature, wind, precipitation),
- Land-use and topographic descriptors,
- Remote sensing indices and hyperspectral measurements.

Examples of classification tasks in this context include determining whether a water sample is contaminated or clean, categorizing air quality into regulatory classes, distinguishing land cover types from satellite imagery, and flagging sensor readings as normal or anomalous.

These problems are challenging because they often involve high-dimensional and correlated predictors, temporal or spatial dependencies that violate independence assumptions, and strong class imbalance since hazardous events typically occur less frequently than safe conditions. To address these challenges, robust preprocessing (imputation of missing values, normalization, encoding of categorical variables) and appropriate validation strategies (e.g., spatial or temporal

block cross-validation) are required to prevent information leakage and to obtain realistic estimates of generalization performance [4].

Beyond predictive accuracy, interpretability and uncertainty quantification are crucial. Feature importance analyses ensure that models rely on ecologically meaningful drivers (e.g., nutrient loads, temperature anomalies), while ensemble dispersion and prediction intervals provide measures of uncertainty that support risk-aware decision making. Finally, recurring domain shifts (due to seasonal cycles, sensor upgrades, or novel events) necessitate periodic retraining, transfer learning, and continuous evaluation to maintain reliable performance over time [4, 5, 7].

Types of algorithms

Several families of classification algorithms are commonly applied in environmental analytics:

- Logistic Regression: a linear probabilistic model providing interpretable coefficients for binary or multi-class outcomes.
- Support Vector Machines (SVMs): margin-based classifiers that use kernels to separate classes in linear or non-linear spaces.
- Decision Trees: hierarchical models that split data into classes using sequential feature-based thresholds.
- Random Forests and Gradient Boosting: ensembles of decision trees that increase predictive power and reduce variance.
- Convolutional Neural Networks (CNNs): deep learning models that automatically extract hierarchical features from imagery or spectral data.

Evaluation Metrics

Evaluating classification models in environmental analytics requires more than overall accuracy, since ecological hazards are often rare and datasets highly imbalanced.

Accuracy

- Proportion of correctly classified samples among all samples.
- Useful when classes are balanced.
- Limitation: Misleading in imbalanced data (e.g., 95% safe vs. 5% hazardous: a trivial “always safe” classifier already achieves 95% accuracy).

Precision

- Fraction of predicted positives that are true positives.
- High precision means few false alarms.
- Important when false positives are costly (e.g., unnecessary remediation efforts).

Recall (Sensitivity)

- Fraction of actual positives that are correctly identified.
- High recall means few missed hazards.
- Crucial in environmental monitoring where false negatives (e.g., missed oil spills, undetected contamination) can have severe impacts.

F1-Score

- Harmonic mean of precision and recall.
- Balances the trade-off between detecting hazards and avoiding false alarms.
- Widely used in imbalanced datasets to summarize performance.

ROC–AUC (Receiver Operating Characteristic)

- Measures trade-off between true positive rate and false positive rate across thresholds.
- Provides a global view of classifier performance.
- Limitation: Can be overly optimistic in highly imbalanced settings.

PR–AUC (Precision–Recall)

- Captures the relationship between precision and recall across thresholds.
- More informative than ROC–AUC when anomalies are rare.
- Especially important in ecological risk detection, where hazardous events are sparse but high-impact.

Environmental problems

Many core environmental challenges can be addressed as classification problems, with different algorithms offering complementary strengths [4, 5].

Water systems

Tasks: harmful algal bloom detection, contamination categorization, trophic state classification.

Suitable algorithms:

- Logistic Regression: interpretable for chemical/physical predictors [2];

- Limitation: weak with non-linear effects.
- Random Forests/Gradient Boosting: capture complex interactions in mixed hydro-chemical datasets [4];

Limitation: reduced interpretability.

Atmospheric systems

Tasks: air quality categorization, detection of industrial pollution plumes [4].

Suitable algorithms:

- SVMs: effective for high-dimensional sensor data [4];
  Limitation: computational cost at scale.
- Gradient Boosting: strong predictive performance with imbalanced air quality data [5];
  Limitation: requires careful tuning.

Forestry and land use

Tasks: wildfire risk mapping, vegetation stress classification, deforestation detection [4, 5].

Suitable algorithms:

- Decision Trees: simple and interpretable for vegetation thresholds;
  Limitation: prone to overfitting.
- Random Forests: robust performance on heterogeneous land-use predictors [5];
  Limitation: less transparent decision logic.
- CNNs: ideal for remote sensing imagery of vegetation/land cover;
  Limitation: data-hungry.

Ecosystem health

Tasks: detection of invasive species from eDNA, classification of wildlife disease risk categories [5, 7].

Suitable algorithms:

- SVMs: well-suited for high-dimensional genomic or eDNA data [4];
  Limitation: less interpretable.
- Logistic Regression: interpretable baseline for disease risk factors [4];

Limitation: weaker performance with complex interactions.

Across these domains, algorithm choice balances predictive accuracy, interpretability, and computational efficiency, with ensemble and deep learning models excelling on complex, high-dimensional data [5], while linear and tree-based models provide transparency and robustness in simpler tasks [4].

Detection of dangerous/anomalous objects

Environmental anomalies often correspond to rare, unexpected, and spatially localized events that pose significant ecological risks. Examples include sudden spikes in particulate matter concentration, abrupt harmful algal blooms, the spread of invasive species, or the appearance of microplastics and heavy metals in aquatic and terrestrial systems. A key challenge is extreme class imbalance: hazardous events may account for less than 5% of observations, making them difficult to detect with conventional classifiers. Moreover, anomalies are often site-specific, so models trained in one region may not generalize to another without careful validation.

List of methods

Several approaches are employed to detect anomalies in environmental data:

- Statistical outlier detection: identifies extreme values under distributional assumptions.
- Resampling and cost-sensitive learning: rebalances class distributions using oversampling, undersampling, or assigning higher penalties to false negatives.
- Ensemble learning: random forests, gradient boosting, and stacked models combine diverse learners to improve robustness against noise.
- Deep anomaly detection: convolutional or autoencoder-based models capture non-linear patterns and highlight unusual spatial or spectral signatures.

Examples of specific methods

Three representative techniques illustrate how different approaches address anomaly detection in practice:

- One-Class SVM: Learns a decision boundary that encloses “normal” data, flagging samples outside this region as anomalies.
  Advantages: effective in high-dimensional spaces, robust to moderate noise.

Limitations: sensitive to kernel choice and hyperparameters; computationally expensive on large datasets.

Applications: abnormal air quality episode detection, identification of invasive species from eDNA samples [6, 7].

- Random Forest: An ensemble of decision trees that captures complex feature interactions and highlights outliers based on classification confidence.

  Advantages: scalable, robust to overfitting, able to handle heterogeneous inputs.

  Limitations: less interpretable than simple trees; performance depends on tuning and sample balance.

  Applications: detecting contaminated soils from chemical assays, identifying unusual water quality profiles [3].
- Autoencoders: Neural networks trained to reconstruct inputs; anomalies are detected when reconstruction error is high.

  Advantages: capture complex non-linear dependencies, suitable for continuous monitoring.

  Limitations: require sufficient “normal” data for training, computationally demanding, less interpretable.

  Applications: real-time anomaly detection in water quality monitoring [1].

Case study: Species distribution risk analysis using GBIF occurrence data and WorldClim bioclimatic predictors

Objective

This case study illustrates a reproducible workflow for environmental risk detection based on species distribution modeling (SDM). The goal is to identify environmental conditions associated with the presence of a target species and to build a machine learning model that estimates the probability of occurrence across a region of interest. In an ecological risk framing, high predicted suitability areas can be interpreted as priority zones for monitoring and prevention, especially for invasive species or disease vectors.

The workflow is implemented in Python using only standard data science libraries and openly available data sources.+

Libraries used

We will use a minimal set of widely used Python libraries for data retrieval, data manipulation, raster sampling, machine learning, and visualization. The chosen stack is intentionally simple and based on “classical” packages commonly used in applied data science. Below the list and purpose of the libraries used:

- **requests -** is used to access web APIs (GBIF) and download climate layers (WorldClim).
- **pandas** and **numpy -** are used for data cleaning, filtering, and feature engineering.
- **rasterio -** is used to read GeoTIFF rasters and extract environmental variables at geographic coordinates.
- **scikit-learn** - provides classification models and evaluation metrics.
- **matplotlib -** is used for simple plots and model interpretation.

This choice supports transparency and makes the workflow easy to run in standard environments such as Jupyter Notebook or Google Colab.

Data source

In this practical tutorial we focus on ***Vespa velutina*** (the Asian hornet), an invasive species that has spread in several European countries and can threaten local biodiversity and beekeeping, because it preys on pollinators such as honeybees. For this reason, detecting where the species is likely to occur next is a practical example of environmental risk detection. The goal of the tutorial is to build a simple machine learning model that learns the environmental conditions associated with the species presence and then produces a habitat suitability or risk score for new locations. More specifically, the dataset consists of presence records (latitude/longitude coordinates and observation metadata) for *Vespa velutina* within the selected region and time window (2000–2024), downloaded through the GBIF public API. To construct the input features, each occurrence point of the species is enriched with climate predictors extracted from WorldClim v2.1 bioclimatic variables, which are global raster layers summarizing long-term temperature and precipitation patterns commonly used in species distribution modeling. The final dataset therefore links each location to a set of climatic indicators (e.g., annual mean temperature, precipitation seasonality) and enables a binary classification task based on presence vs pseudo-absence (background) points sampled in the same region.

Data sources: GBIF Occurrence API https://techdocs.gbif.org/en/openapi/v1/occurrence and WorldClim v2.1 Bioclim https://www.worldclim.org/data/worldclim21.html

Data types

The tutorial combines two common types of geospatial data that are frequently used together in environmental analytics: *occurrence point data* (where something was observed) and *raster data* (spatial layers describing environmental conditions everywhere in the region).

- **Occurrence data** (point observations): The first dataset is a table of GBIF occurrence records for Vespa velutina. In the code, each observation is stored as one row with geographic coordinates (decimalLatitude, decimalLongitude) and basic metadata: gbifID, year, basisOfRecord, datasetKey, and (when available) coordinateUncertaintyInMeters These points represent presence locations (label 1), meaning places where the species was actually reported. Since GBIF provides mainly presence-only data, these records do not directly include confirmed absences.

```
# --------------------------------
SPECIES_NAME = "Vespa velutina"  # invasive Asian hornet example
# Italy bounding box (lon_min, lat_min, lon_max, lat_max)
ROI = dict(lon_min=6.6, lat_min=36.6, lon_max=18.6, lat_max=47.2)

YEAR_FROM, YEAR_TO = 2000, 2024

taxon_key, match_json = gbif_taxon_key(SPECIES_NAME)

pres = fetch_gbif_occurrences(
    taxon_key, ROI, YEAR_FROM, YEAR_TO,
    max_points=6000
)

pres = pres.dropna(subset=["decimalLatitude", "decimalLongitude"]).copy()
pres = pres.drop_duplicates(subset=["decimalLatitude", "decimalLongitude"]).copy()

pres["label"] = 1
pres.head()
```

1 to 5 of 5 entries Filter

index	gbifID	species	scientificName	year	decimalLatitude	decimalLongitude	basisOfRecord	datasetKey	coordinateUncertaintyInMeters	label
0	4510393598	Vespa velutina	Vespa velutina Lepeletier, 1836	2024	46.551558	6.637408	HUMAN_OBSERVATION	50c9509d-22c7-4a22-a47d-8c48425ef4a7	5.0	1
1	5219609581	Vespa	Vespa velutina	2024	47.01748	6.94541	HUMAN_OBSERVATION	81981d98-e27f-4155-	3535.0	1

- **Raster data** (gridded environmental layers): The second dataset consists of WorldClim v2.1 bioclimatic rasters, stored as GeoTIFF files. Each raster is a grid covering the region, where every cell contains a climatic value. In this tutorial we use six predictors: BIO1, BIO4, BIO5, BIO12, BIO14, BIO15, which will become the columns bio1, bio4, bio5, bio12, bio14, and bio15. Raster values are extracted using rasterio.

```
# WorldClim bioclim variables to use.
# https://www.worldclim.org/data/bioclim.html
# BIO1: Annual Mean Temperature
# BIO4: Temperature Seasonality (standard deviation ×100)
# BIO5: Max Temperature of Warmest Month
# BIO12: Annual Precipitation
# BIO14: Precipitation of Driest Month
# BIO15: Precipitation Seasonality (Coefficient of Variation)
BIO_VARS = [1, 4, 5, 12, 14, 15]

# We can download where we will be downloading our data
WC_URL = "https://geodata.ucdavis.edu/climate/worldclim/2_1/base/wc2.1_2.5m_bio.zip"

# ... download and unzip ...we need to keep only the files related to the BIO_VARS

bio_files = {}
for k in BIO_VARS:
    path = os.path.join(extract_dir, f"wc2.1_2.5m_bio_{k}.tif")
    bio_files[k] = path

bio_files
```

```
{1: 'data_sdm/wc2.1_2.5m_bio/wc2.1_2.5m_bio_1.tif',
 4: 'data_sdm/wc2.1_2.5m_bio/wc2.1_2.5m_bio_4.tif',
 5: 'data_sdm/wc2.1_2.5m_bio/wc2.1_2.5m_bio_5.tif',
 12: 'data_sdm/wc2.1_2.5m_bio/wc2.1_2.5m_bio_12.tif',
 14: 'data_sdm/wc2.1_2.5m_bio/wc2.1_2.5m_bio_14.tif',
 15: 'data_sdm/wc2.1_2.5m_bio/wc2.1_2.5m_bio_15.tif'}
```

- How the two data types are combined (point-to-raster extraction): The integration step is performed by enriching each GBIF point with raster values: for each coordinate (lon,lat)(lon, lat)(lon,lat), we read the value from each WorldClim raster and append it as a new feature column. This produces a standard machine learning table where each row is a location and each predictor is a climatic variable.

```
def sample_raster_values(raster_path, lons, lats):
    values = np.empty(len(lons), dtype=float)
    with rasterio.open(raster_path) as src:
        nodata = src.nodata
        for i, (x, y) in enumerate(zip(lons, lats)):
            v = next(src.sample([(x, y)]))[0]
            values[i] = np.nan if (nodata is not None and v == nodata) else float(v)
    return values

def add_bioclim(df, bio_files):
    out = df.copy()
    lons = out["decimalLongitude"].to_numpy()
    lats = out["decimalLatitude"].to_numpy()
    for k, path in bio_files.items():
        out[f"bio{k}"] = sample_raster_values(path, lons, lats)
    return out

pres_feat = add_bioclim(pres, bio_files)
pres_feat.head()
```

1 to 5 of 5 entries Filter

index	gbifID	species	scientificName	year	decimalLatitude	decimalLongitude	basisOfRecord	datasetKey	coordinateUncertaintyInMeters	label	bio1	bio4	bio5
0	4510393598	Vespa velutina	Vespa velutina Lepeletier, 1836	2024	46.551558	6.637408	HUMAN_OBSERVATION	50c9509d-22c7-4a22-a47d-8c48425ef4a7	5.0	1	8.567000389099121	648.1673583984375	23.148000717163086
1	5219609581	Vespa velutina	Vespa velutina Lepeletier, 1836	2024	47.01748	6.94541	HUMAN_OBSERVATION	81981d98-e27f-4155-9a94-9eae9bbad2be	3535.0	1	7.342833518981934	627.8450927734375	21.215999603271484

Since GBIF provides mainly presence-only records, the tutorial creates **background points** in the same region and assigns them label 0. These points are used as a proxy for non-presence locations, allowing a binary classification dataset. These points are used as a proxy for non-presence locations, allowing a binary classification dataset.

```
bg = random_points_bbox(n_bg, ROI, seed=42)
bg["label"] = 0

bg_feat = add_bioclim(bg, bio_files)
bg_feat.head()
```

	decimalLongitude	decimalLatitude	label	bio1	bio4	bio5	bio12	bio14	bio15
0	15.887473	46.851860	0	9.194166	768.208618	24.972	842.0	30.0	38.745869
1	11.866541	43.871469	0	10.660334	650.370911	25.180	1430.0	64.0	30.592279
2	16.903175	37.555253	0	NaN	NaN	NaN	NaN	NaN	NaN
3	14.968416	43.947677	0	NaN	NaN	NaN	NaN	NaN	NaN
4	7.730128	42.998380	0	NaN	NaN	NaN	NaN	NaN	NaN

Data cleaning

Occurrence datasets often require a basic cleaning step because they may contain duplicate coordinates (the same location reported multiple times), uncertain coordinates with low spatial precision, and, in rare cases, points that fall outside the selected region of interest. Since the goal of this tutorial is to associate environmental conditions with species presence, inaccurate or repeated records could bias the analysis and the resulting machine learning model. For this reason, the workflow applies standard filters by removing records with missing latitude or longitude, keeping only observations within the ROI bounding box, removing duplicate coordinate pairs, and, when available, optionally excluding points with very high coordinate uncertainty.

Final machine learning table for training

The presence points and background points (both with extracted BIO predictors) are combined into one dataset used for training. The input feature matrix X contains the selected bioclimatic variables extracted from WorldClim (BIO predictors), while the target vector y contains the binary labels indicating presence versus pseudo-absence. This produces a standard supervised learning dataset suitable for training classification models.

```
# ------------------------------
# 7) Build final ML table (X,y)
# ------------------------------
feature_cols = [f"bio{k}" for k in BIO_VARS]
cols = ["label"] + feature_cols

data = pd.concat([pres_feat[cols], bg_feat[cols]], ignore_index=True)
data = data.dropna().copy()

X = data[feature_cols].to_numpy()
y = data["label"].to_numpy()

data["label"].value_counts()
```

label	count
0	3778
1	1566

dtype: int64

Train/test split and baseline model (Logistic Regression)

To evaluate the model's ability to generalize to unseen data, the dataset is split into training and test sets. A stratified split is used to preserve the proportion of presence and pseudo-absence points in both subsets, ensuring that performance estimates are not biased by class imbalance. **Logistic Regression** is used as a first baseline model because it is simple and interpretable, provides probabilistic predictions, and is a strong reference method for binary classification problems. **Evaluation metrics:** Because ecological presence datasets are often imbalanced, model performance is assessed using metrics that are informative beyond accuracy, including ROC-AUC, precision and recall, F1-score, and the confusion matrix. This supports a more meaningful interpretation of predictive performance in an environmental risk detection context.

```python
# ------------------------------
# 8) Train/test split + Logistic Regression
# ------------------------------
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=0, stratify=y
)

logreg = LogisticRegression(max_iter=3000, class_weight="balanced")
logreg.fit(X_train, y_train)

proba_lr = logreg.predict_proba(X_test)[:, 1]
pred_lr = (proba_lr >= 0.5).astype(int)

print("LogReg ROC-AUC:", roc_auc_score(y_test, proba_lr))
print(classification_report(y_test, pred_lr, digits=3))
print("Confusion matrix:\n", confusion_matrix(y_test, pred_lr))
```

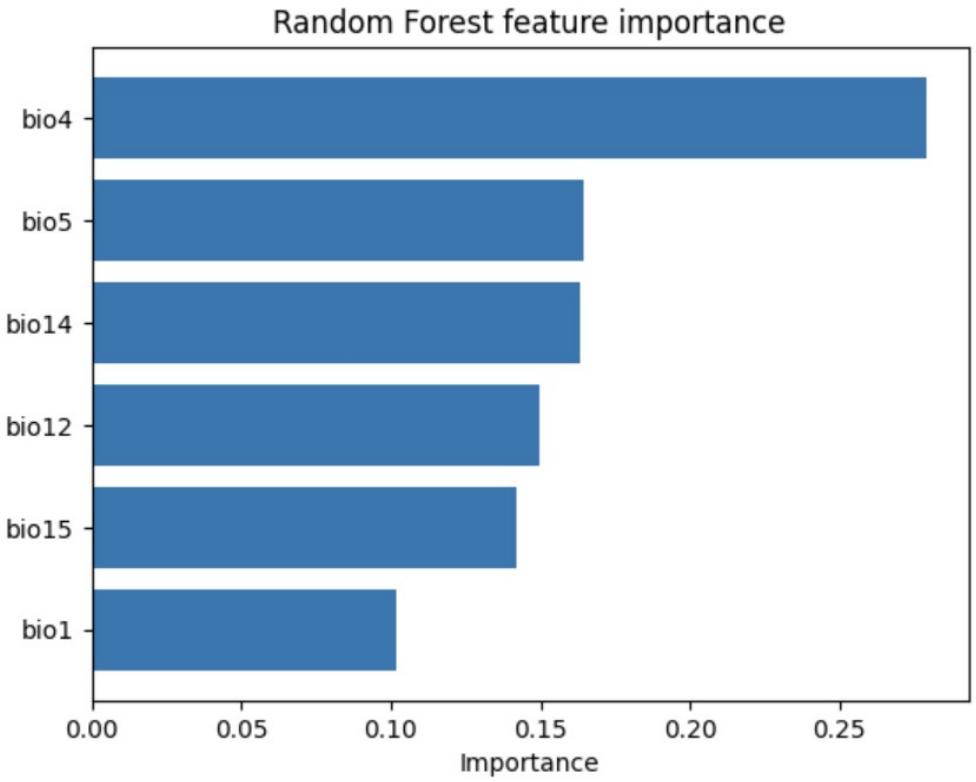
```
LogReg ROC-AUC: 0.9246512131422616
              precision    recall  f1-score   support

           0      0.952     0.820     0.881       945
           1      0.674     0.900     0.771       391

    accuracy                          0.844      1336
   macro avg      0.813     0.860     0.826      1336
weighted avg      0.871     0.844     0.849      1336

Confusion matrix:
 [[775 170]
 [ 39 352]]
```

Random Forest and feature importance

Random Forest is introduced as a second model because species–environment relationships are often nonlinear and may involve interactions between predictors. Random Forest typically performs well on tabular environmental datasets and is robust to complex decision boundaries. Random Forest also provides a practical form of interpretability through feature importance, which highlights which climatic predictors contribute most to separating presence from pseudo-absence points. This supports ecological interpretation, for example by identifying associations with warmer conditions, sensitivity to dry-season precipitation, or the role of seasonality constraints. Overall, this step reinforces an “explainable risk detection” perspective.

```
# ------------------------------
# 9) Random Forest + feature importance
# ------------------------------
rf = RandomForestClassifier(
    n_estimators=400,
    random_state=0,
    class_weight="balanced_subsample",
    n_jobs=-1
)
rf.fit(X_train, y_train)

proba_rf = rf.predict_proba(X_test)[:, 1]
pred_rf = (proba_rf >= 0.5).astype(int)

print("RF ROC-AUC:", roc_auc_score(y_test, proba_rf))
print(classification_report(y_test, pred_rf, digits=3))
print("Confusion matrix:\n", confusion_matrix(y_test, pred_rf))

# Importance plot
importances = rf.feature_importances_
order = np.argsort(importances)

plt.figure()
plt.barh([feature_cols[i] for i in order], importances[order])
plt.title("Random Forest feature importance")
plt.xlabel("Importance")
plt.show()
```

```
RF ROC-AUC: 0.988926778440845
              precision    recall  f1-score   support

           0      0.970     0.972     0.971       945
           1      0.933     0.928     0.931       391

    accuracy                          0.960      1336
   macro avg      0.952     0.950     0.951      1336
weighted avg      0.960     0.960     0.960      1336

Confusion matrix:
 [[919  26]
 [ 28 363]]
```



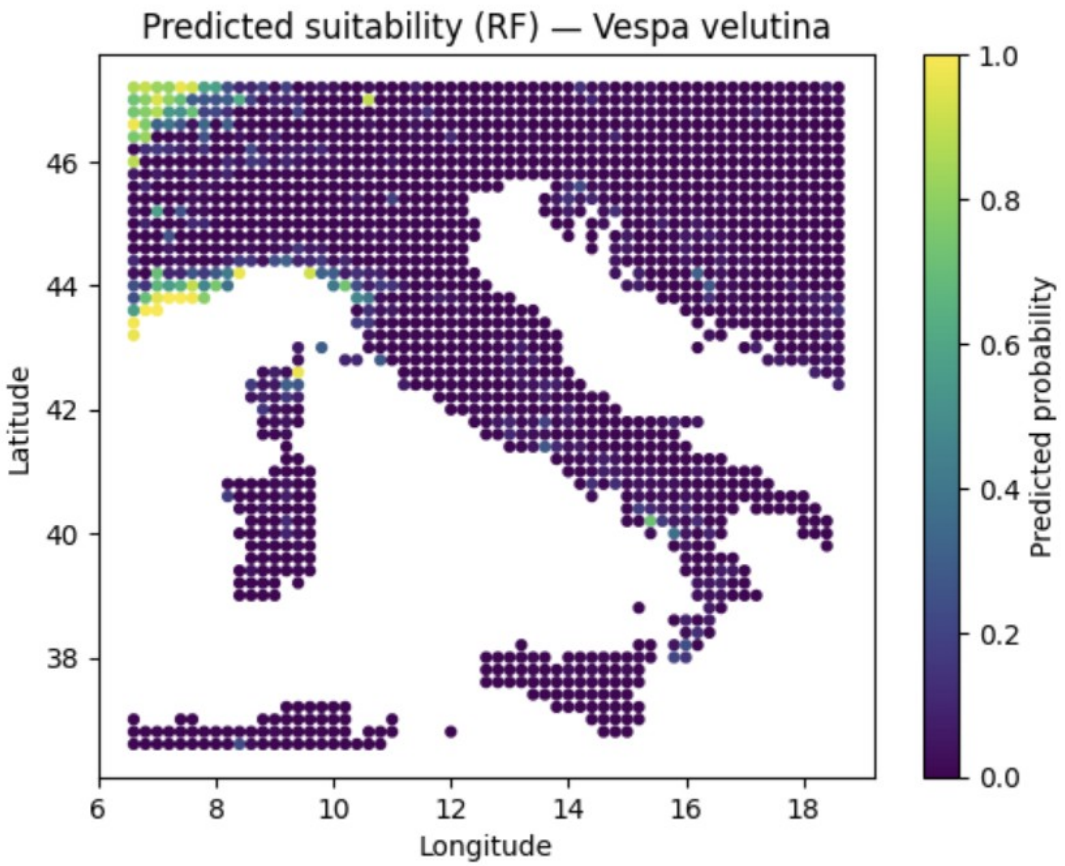
The Random Forest model achieves very strong performance on this dataset, with a ROC–AUC of 0.99 and an overall accuracy of 0.96, indicating that the climatic predictors provide a clear separation between presence and pseudo-absence points. The confusion matrix shows only a limited number of errors (26 false positives and 28 false negatives), and the classification report confirms high precision and recall for both classes. Looking at the feature importance plot, BIO4 (temperature seasonality) is the most influential variable, suggesting that the variability of temperature across the year is a key factor in distinguishing suitable vs unsuitable conditions for Vespa velutina. Other relevant predictors include BIO5 (maximum temperature of the warmest

month) and precipitation-related variables (BIO12, BIO14, BIO15), while BIO1 (annual mean temperature) appears less informative on its own. These results support the idea that *Vespa velutina* suitability is driven not only by average climate levels, but also by seasonality and extremes, which is consistent with ecological expectations for invasive species establishment.

Spatial prediction

A key advantage of species distribution modeling is that the trained model can be applied to unsampled locations. To illustrate this, a grid of points is generated across the region of interest and scored using the fitted classifier, producing a spatial prediction surface.The predicted probability can be interpreted as a habitat suitability score, a proxy for probability of establishment, or a monitoring priority indicator. High predicted values correspond to areas where the species could plausibly persist, even if no occurrences have been reported there yet.

```
# --------------------------------
# 10) Optional: quick suitability map on a coarse grid
#     (no geopandas; just a scatter heatmap)
# --------------------------------
def make_grid(roi, step_deg=0.2):
    lons = np.arange(roi["lon_min"], roi["lon_max"] + 1e-9, step_deg)
    lats = np.arange(roi["lat_min"], roi["lat_max"] + 1e-9, step_deg)
    grid = [(x, y) for y in lats for x in lons]
    return pd.DataFrame(grid, columns=["decimalLongitude", "decimalLatitude"])

grid = make_grid(ROI, step_deg=0.2)
grid_feat = add_bioclim(grid, bio_files)
grid_feat = grid_feat.dropna(subset=feature_cols).copy()

Xg = grid_feat[feature_cols].to_numpy()
grid_feat["suitability"] = rf.predict_proba(Xg)[:, 1]

plt.figure()
plt.scatter(grid_feat["decimalLongitude"], grid_feat["decimalLatitude"], c=grid_feat["suitability"], s=12)
plt.title(f"Predicted suitability (RF) — {SPECIES_NAME}")
plt.xlabel("Longitude")
plt.ylabel("Latitude")
plt.colorbar(label="Predicted probability")
plt.show()
```



Exercise Tasks

- Repeat the same workflow of this tutorial for a different species of your choice (for example an invasive species, a pollinator, or a species relevant for your region).
- Replace SPECIES_NAME in the code, change the ROI for your region and keep the same time window (2000–2024), and run the full pipeline: download GBIF occurrences (presence points), clean the coordinates, generate pseudo-absence points, extract the same WorldClim predictors (bio1, bio4, bio5, bio12, bio14, bio15), and train both Logistic Regression and Random Forest. To select a suitable species, make sure your GBIF query returns enough records (at least ~200 points after cleaning).
- As a short report, include: (i) the number of presence points used, (ii) ROC–AUC and confusion matrix for both models, and (iii) the Random Forest feature importance plot.

- Finally, answer these questions: Which model performs better and why? Which BIO variable is the most important for your species? Do the results suggest that the species is driven more by temperature-related or precipitation-related constraints?

Assessment

What is the main objective of applying classification models in environmental risk detection?

A. Predict continuous environmental variables such as temperature or pollutant concentration
B. Group environmental observations into unlabeled clusters
C. Assign observations to predefined risk categories (e.g. hazardous vs. safe)
D. Reduce the dimensionality of environmental datasets

Why is overall accuracy often insufficient for evaluating classification models in environmental applications?

A. Because environmental data are always noisy
B. Because accuracy cannot be computed for multi-class problems
C. Because environmental datasets are often highly imbalanced, with rare hazardous events
D. Because accuracy ignores model interpretability

Which evaluation metric is most critical when the goal is to minimize missed environmental hazards (false negatives)?

A. Precision
B. Accuracy
C. Recall (Sensitivity)
D. ROC–AUC

Which algorithm is particularly well-suited for detecting oil spills or marine litter in satellite or drone imagery?

A. Logistic Regression
B. Support Vector Machines
C. Random Forests

D. Convolutional Neural Networks (CNNs)

What best describes the role of classification model outputs in environmental risk management?

A. They replace expert judgment entirely
B. They provide exact forecasts of future environmental conditions
C. They support decision-making by identifying and prioritizing risk categories
D. They eliminate uncertainty in monitoring systems

Additional Sources

Introductory tutorials on forecasting with Python (e.g., scikit-learn, pandas).

Machine Learning - A First Course for Engineers and Scientists, Andreas Lindholm, Niklas Wahlström, Fredrik Lindsten, and Thomas B. Schön

Deep Learning with Python, Second Edition, François Chollet

Applications of Deep Neural Networks with Keras, Jeff Heaton

Understanding Machine Learning: From Theory to Algorithms, Shai Shalev-Shwartz and Shai Ben-David

Understanding Machine Learning: From Theory to Algorithms, Shai Shalev-Shwartz and Shai Ben-David

Bibliography

[1] El-Shafeiy, E., Alsabaan, M., Ibrahem, M. I., & Elwahsh, H. (2023). Real-time anomaly detection for water quality sensor monitoring based on multivariate deep learning technique. Sensors, 23(20), 8613.

[2] Hu, X., Luo, H., Guo, M., & Wang, J. (2022). Ecological technology evaluation model and its application based on Logistic Regression. Ecological Indicators, 136, 108641.

[3] Madani, A., Hagage, M., & Elbeih, S. F. (2022). Random Forest and Logistic Regression algorithms for prediction of groundwater contamination using ammonia concentration. Arabian Journal of Geosciences, 15(20), 1619.

[4] Maxwell, A. E., Warner, T. A., & Fang, F. (2018). Implementation of machine-learning classification in remote sensing: An applied review. International Journal of Remote Sensing, 39 (9), 2784–2817.

[5] Pichler, M., & Hartig, F. (2023). Machine learning and deep learning—A review for ecologists. Methods in Ecology and Evolution, 14 (4), 994–1016.

[6] Scholkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. Neural Computation, 13(7), 1443–1471.

[7] Zinger, L., Bonin, A., Alsos, I.G., Ba´lint, M., Bik, H., Boyer, F., Chariton, A.A., Creer, S., Coissac, E., Deagle, B.E. & De Barba, M., (2019). DNA metabarcoding—Need for robust experimental designs to draw sound ecological conclusions. Molecular ecology, 28(8), pp.1857-1862.