LECTURE 3

DATA REPRESENTATIONS

PRESS → OR SPACE.

© Marc Rußwurm
Licensed under **Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0)**.
You may share and adapt these slides for teaching, research, and other non-commercial educational purposes with attribution.
Commercial training, paid workshops, consulting seminars, or incorporation into commercial course products require prior permission.

UNIVERSITÄT BONN

Learning outcomes

Understand the classic machine learning pipeline: feature design followed by a classification model.

Not comfortable — Very comfortable

Click to respond

Understand the curse of dimensionality and why we select features.

Not comfortable — Very comfortable

Click to respond

Understand how principal component analysis de-correlates data.

Not comfortable — Very comfortable

Click to respond

Know examples of problem-specific features designed with domain knowledge.

Not comfortable — Very comfortable

Click to respond

Understand the mechanisms behind decision trees and random forests.

Not comfortable — Very comfortable

Click to respond

Understand the principle of bias and variance in classification.

Not comfortable — Very comfortable

Click to respond

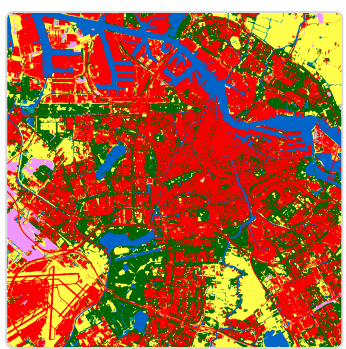
Recap: Geospatial Data

- satellite imagery
- thematic rasters such as land cover
- vector geometries and attributes
- model and reanalysis fields
- observations collected at different times and scales

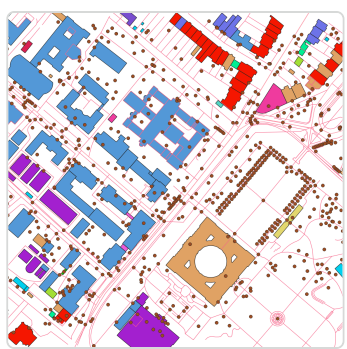
Each source describes locations on its own spatial and temporal support.



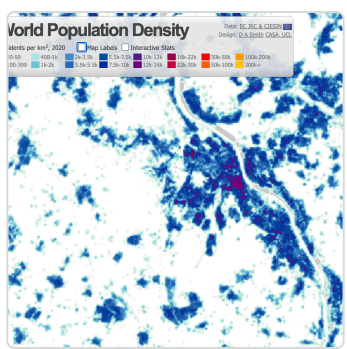
Satellite imagery (Bonn, Sentinel-2)



Thematic raster (land cover, Amsterdam)

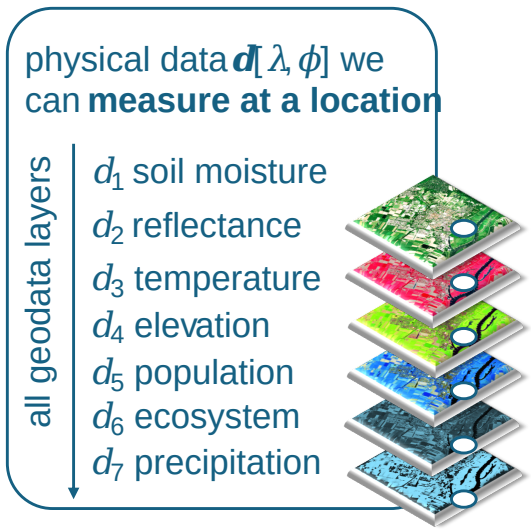


Vector geometries (OpenStreetMap)



Model / reanalysis field (population density)

Represent a Coordinate by Geospatial Data



Sampling all p variables at location i produces a fixed feature vector:

$$\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{ip}]$$

For example:

$$\mathbf{x}_i = [\text{elevation},\ \text{slope},\ \text{spectral bands},\ \text{temperature},\ \text{building density}]$$

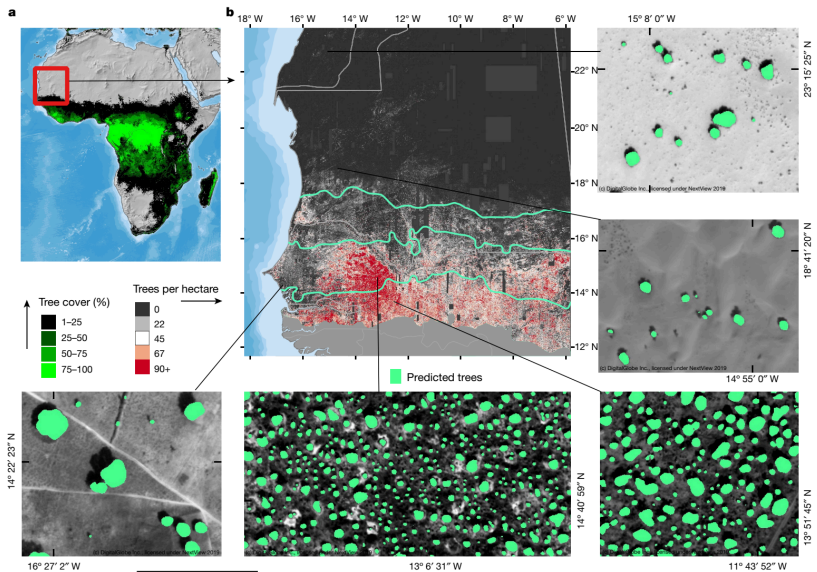
Repeating the extraction at n locations creates a feature matrix:

$$\mathbf{X} \in \mathbb{R}^{n \times p}$$

features x_i → machine-learning model → prediction $\hat{y}_i$

Training target y_i: kept separate from the input features.

Example: Counting Trees Outside Forests



1.8 billion trees

across 1.3 million km² of West African drylands

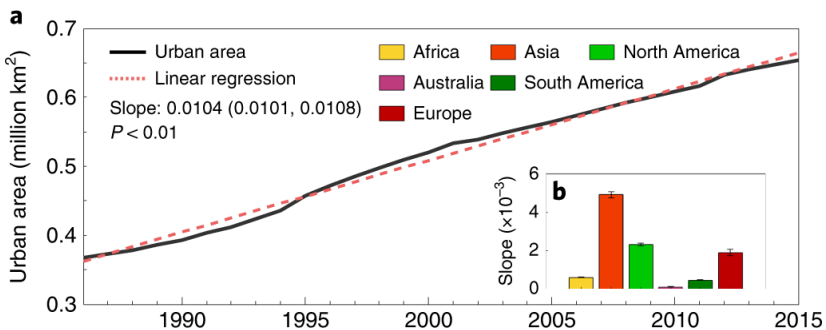
Land cover: isolated trees also occur outside areas classified as forest.

Measurement: 0.5 m satellite imagery and deep learning identify individual crowns larger than 3 m².

A land cover class and an individual-tree count describe different properties of a place.

Brandt et al. (2020), *Nature* 587, 78–82 · An unexpectedly large count of trees in the West African Sahara and Sahel · Fig. 1, caption cropped. Satellite imagery © DigitalGlobe, NextView licence.

Example: Measuring Global Urban Expansion



Global urban area and continental growth rates, 1985–2015

+80% urban area

362,747 to 653,354 km², 1985–2015

Land cover: annual maps distinguish built-up from non-urban surfaces.

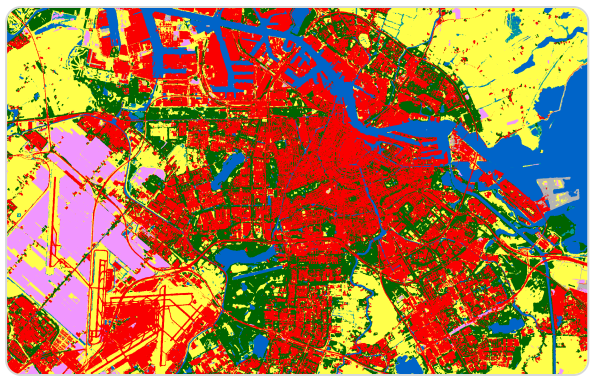
Measurement: 30 m Landsat imagery, random forests and temporal change analysis.

Consistent land cover maps turn pixels into estimates of global change.

Liu et al. (2020), *Nature Sustainability* 3, 564–570 · High-spatiotemporal-resolution mapping of global urban change from 1985 to 2015 · Original Fig. 1.

Example ESA WorldCover

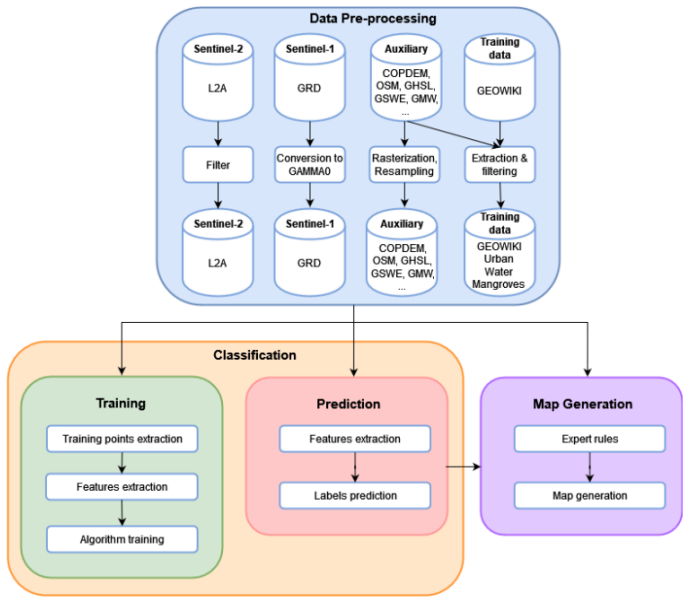
WorldCover classifies each 10 m pixel into one of 11 land cover classes.



ESA WorldCover 2021 · Amsterdam detail

- Tree cover
- Grassland
- Built-up
- Snow and ice
- Herbaceous wetland
- Moss and lichen
- Shrubland
- Cropland
- Bare / sparse vegetation
- Permanent water
- Mangroves

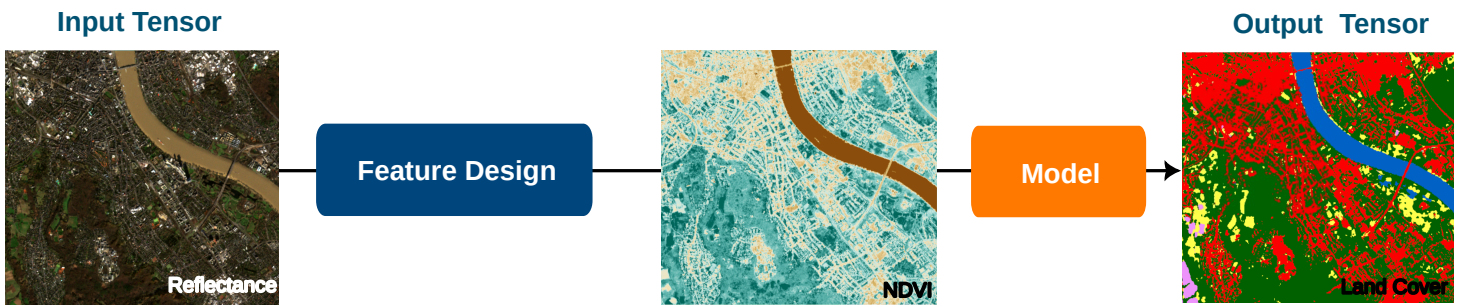
© ESA WorldCover project 2021 · Zanaga et al. (2022)



2020 workflow: Terrascope, ESA WorldCover Product. Open full-size diagram.

Machine Learning Pipeline

Good, hand-selected features combined with a suitable model determine the quality of the result.

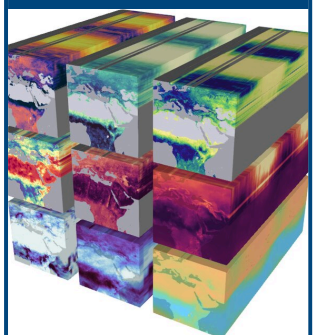


● This lecture · More Data, PCA & Better Features ● This lecture · Decision Trees & Random Forests

Lecture Overview

How to Improve Machine Learning Models?

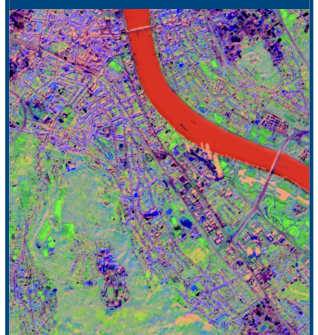
1 · More Data



Curse of Dimensionality

→

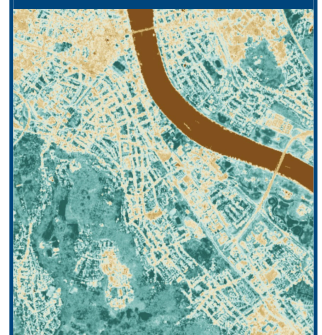
2 · Less Redundant Data



Decorrelation through PCA

→

3 · More Descriptive Data



Domain Knowledge

→

4 · Better (Non-linear) Models



Random Forests

More data and better features improve machine learning – as long as the dimensions stay informative.

More Data - Integrating More Data Dimensions

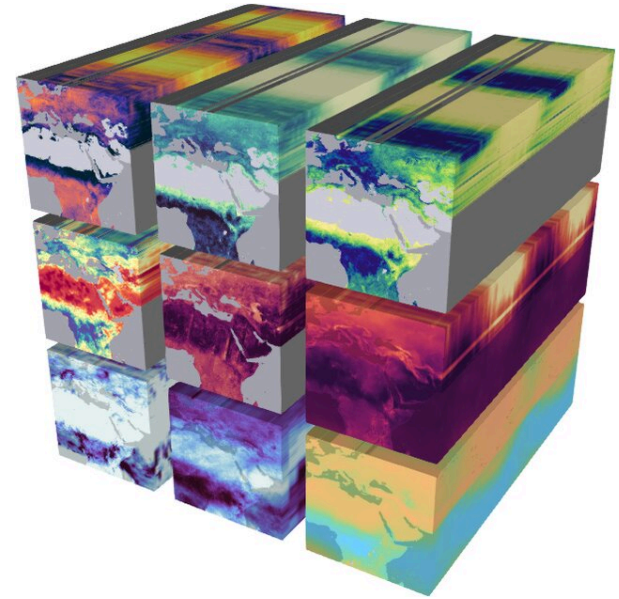
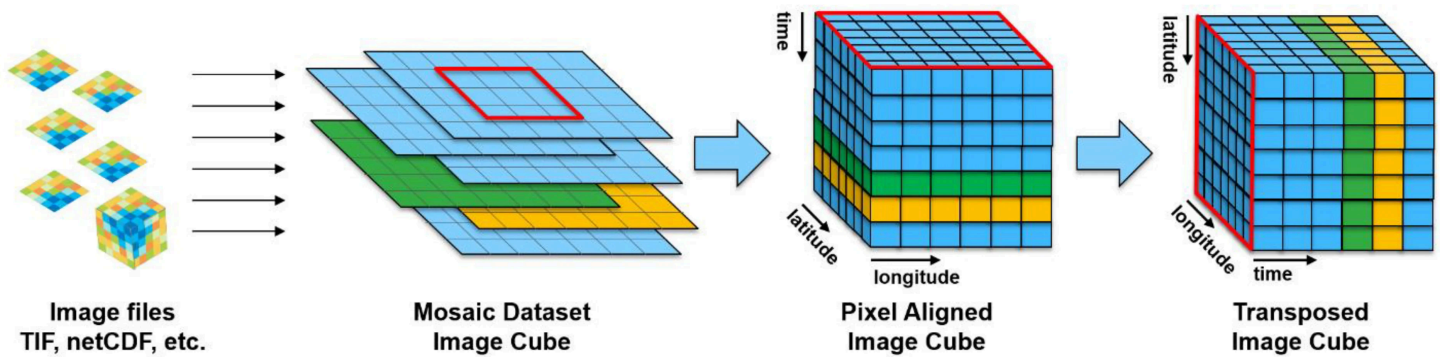


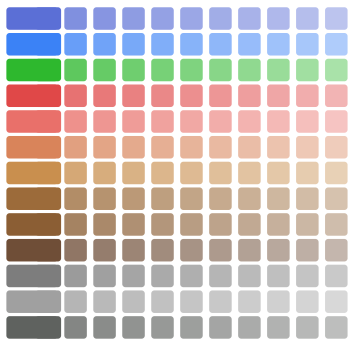
Image by Miguel D. Mahecha

Datacubes align Data Spatially and Temporally

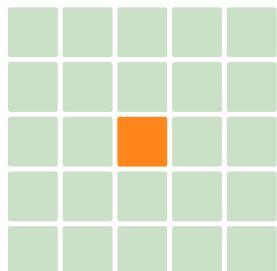


Source: Kopp et al. (2019), *Achieving the Full Vision of Earth Observation Data Cubes*, *Data* 4(3), 94.

Dimensions Add Up Quickly



13 bands × 12 months



5 × 5 neighbourhood



auxiliary layers

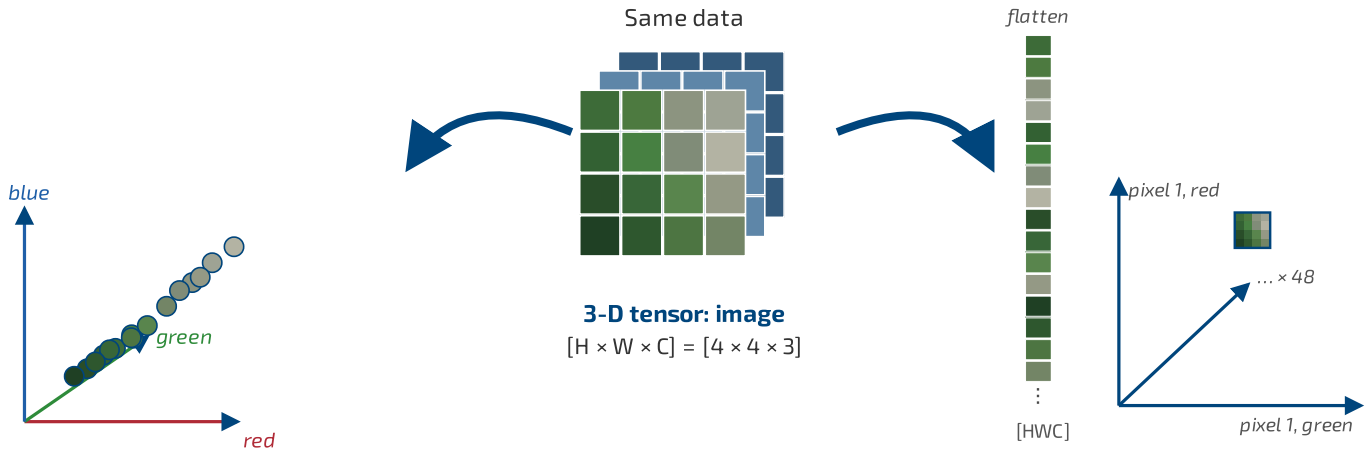
dimensions of one feature vector

4,225

$D = 13 \times 12 \times 25 + 13 \times 25$

1. one Sentinel-2 pixel — **D = 13**
2. + time: one image per month — **D = 156**
3. + local context: 5 × 5 neighbourhood — **D = 3,900**
4. + auxiliary layers (elevation, slope, land-cover one-hot, …) — **D = 4,225**

You Design the Vector Space

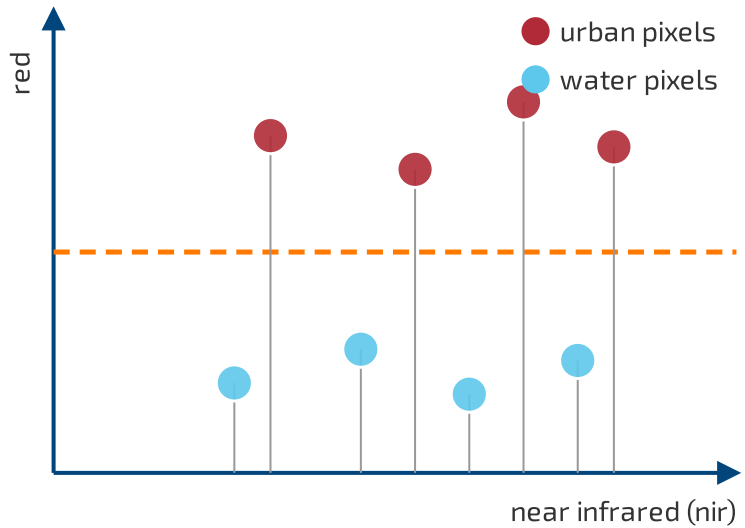


16 data points in 3 dimensions
span a **3-D vector space**

1 data point in 48 dimensions
span a **48-D vector space**

The same data can easily be represented in few or in many dimensions.

Blessing of Dimensionality: More Information



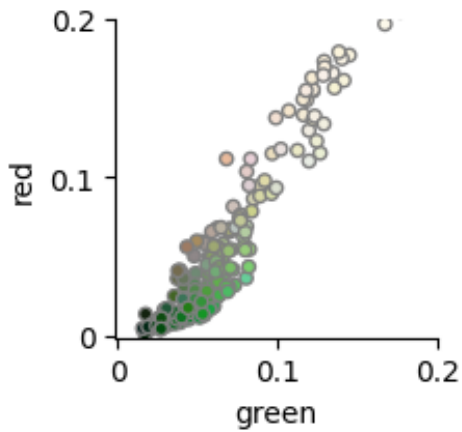
More dimensions can make data separable – good for classification.

Urban and water pixels overlap in the near-infrared band alone.

Data is not separable in nir alone – adding red makes it separable.

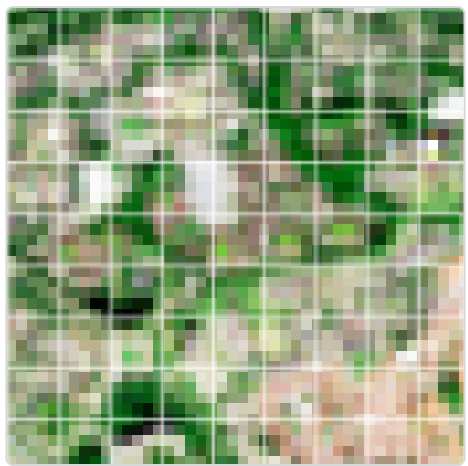
Curse of Dimensionality: Redundancy

Remote sensing bands are highly correlated



Pixels are often bright or dark overall: a very green pixel is usually also very red.

Image pixels are spatially correlated

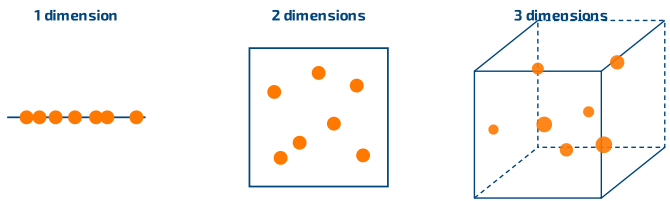


Neighbouring pixels most likely have a very similar colour.

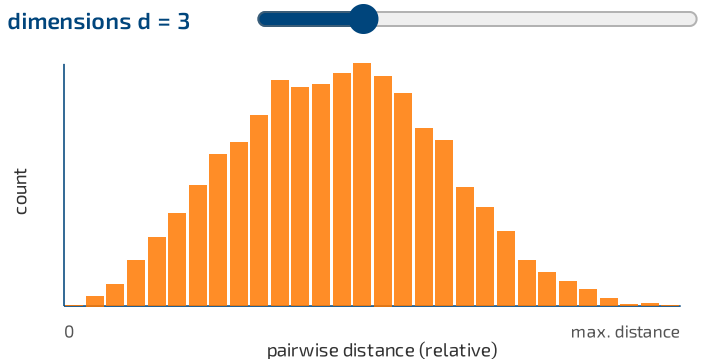
Correlated dimensions add redundancy, not information.

Curse of Dimensionality: Empty Spaces

A higher-dimensional space becomes **emptier** – we need **exponentially more data** to fill it evenly.



In high dimensions, all samples are **similarly far apart**: distances lose their meaning, and classifiers that rely on "closeness" (e.g. nearest neighbours) become less effective.



relative contrast $(d_{max} - d_{min}) / d_{min}$ = **10.80** (150 uniform random points)

Blessing and Curse of Dimensionality

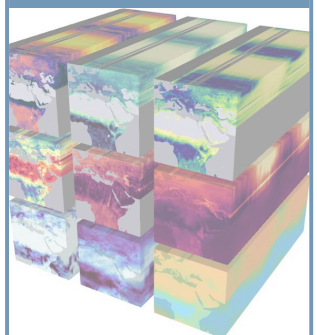
Blessing	Curse
Extra dimensions can provide new, independent information	Correlated (redundant) dimensions add no information
More dimensions help capture complex patterns	More data is needed to fill the emptier high-dimensional space
	Distances lose meaning; classifiers become less effective

Keep as many dimensions as necessary, but as few as possible.

⇒ We need **informative** dimensions.

Lecture Overview

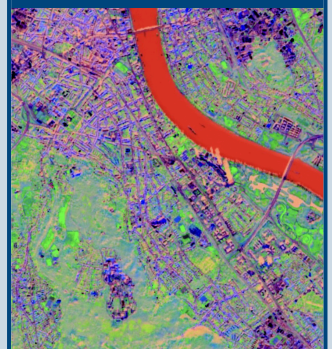
1 · More Data ✓



Curse of Dimensionality

→

2 · Less Redundant Data



Decorrelation through PCA

→

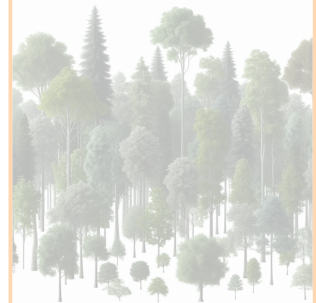
3 · More Descriptive Data



Domain Knowledge

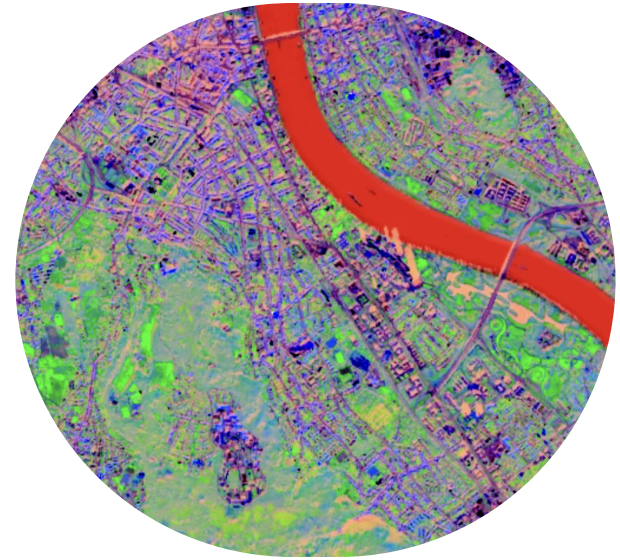
→

4 · Better (Non-linear) Models



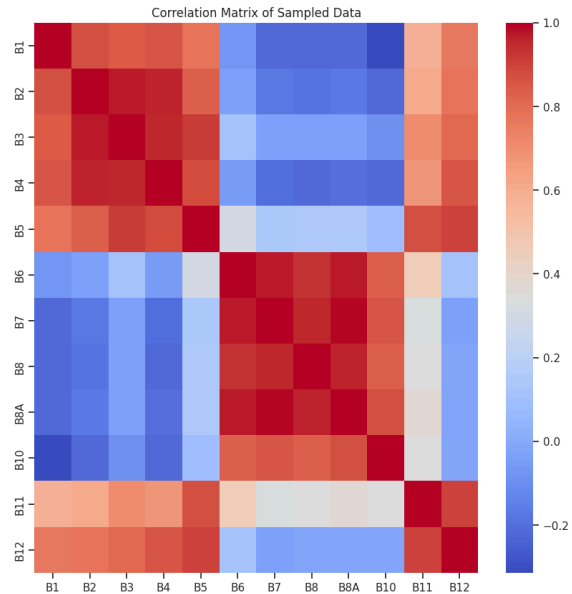
Random Forests

PCA: De-correlating Data



Sentinel-2 scene of Bonn, first three principal components as RGB.

Sentinel-2 Bands Are Highly Correlated



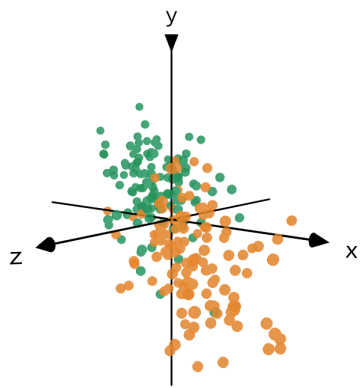
Correlation matrix of 12 Sentinel-2 bands (red = +1, blue = −1)

- **visible** bands (B1–B5) are highly correlated with each other
- visible bands are **not** correlated with **near-infrared** (B6–B9)
- visible bands are again correlated with **SWIR** (B11, B12)

Many of our dimensions carry (almost) the same information.

Recap Lecture 1: PCA

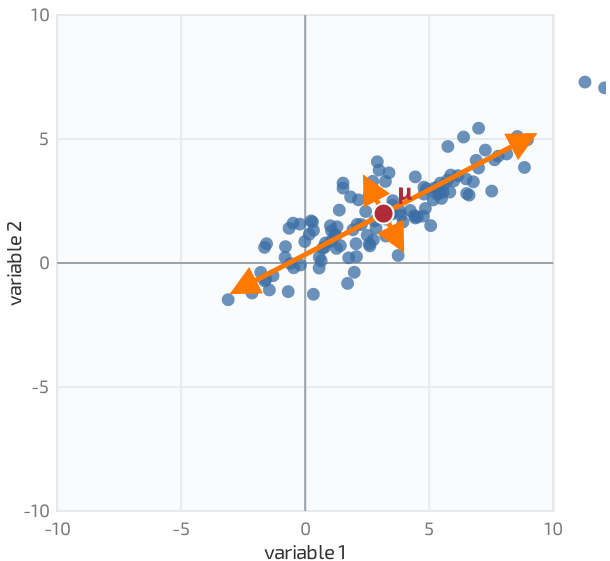
Rotate to PCA projection | Reset | ☑ Show projections



In Lecture 1, we used **Principal Component Analysis (PCA)** to **visualize** high-dimensional vectors.

> **Now: PCA rotates correlated data onto its few most dominant directions.**

PCA to De-correlate Data



correlation ρ 0.85

spread σ_1 3.0

spread σ_2 1.8

1 · translate (center) | 2 · rotate to PCs | reset

☑ show principal axes

Sample covariance matrix C

$$\begin{pmatrix} 9.10 & 4.45 \\ 4.45 & 2.93 \end{pmatrix}$$

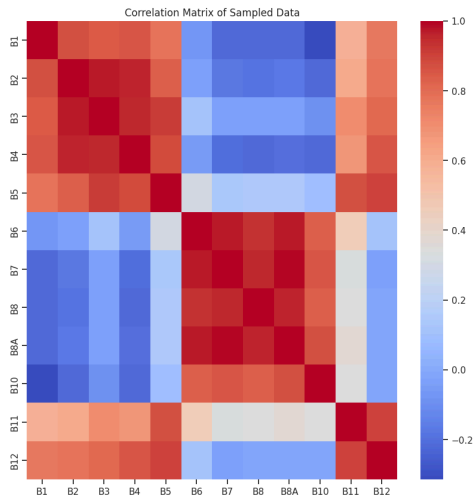
mean $\mu = (3.16, 1.99)$

eigenvalues $\lambda_1 = 11.43$, $\lambda_2 = 0.60$ → PC 1 explains **95%**

PCA **centres** and **rotates** the data so that the covariances vanish. The new axes, the **principal components**, point along the most dominant directions of the data.

PCs Are Uncorrelated “Synthetic Bands”

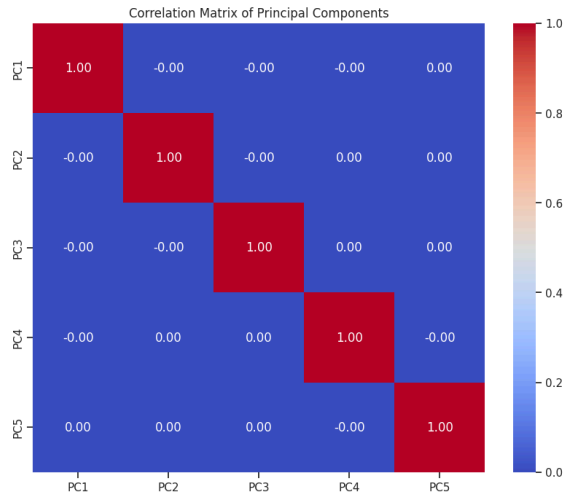
12 correlated Sentinel-2 bands



PCT

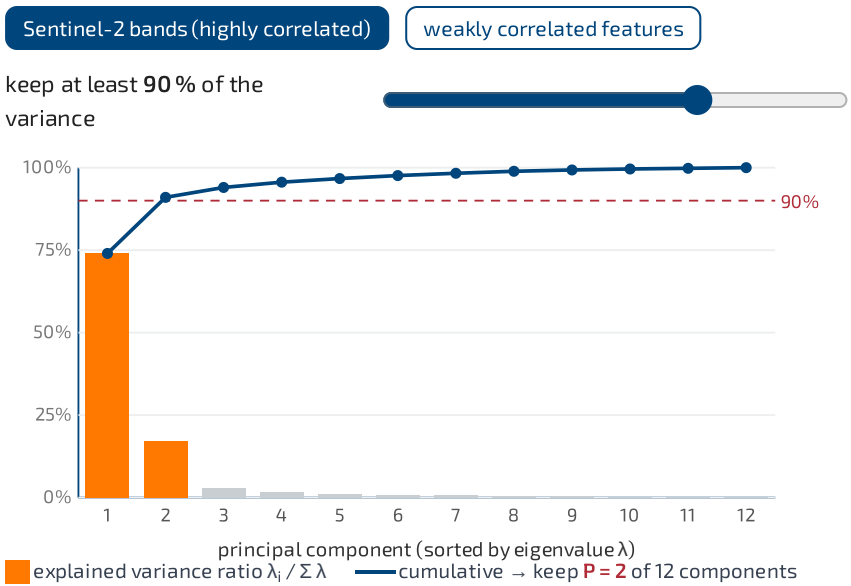
principal component transformation

(fewer) uncorrelated principal components



Same goal as a spectral index – remove redundancy – but found **from the data alone**, without knowing what the bands mean.

Few Directions Carry Most Information



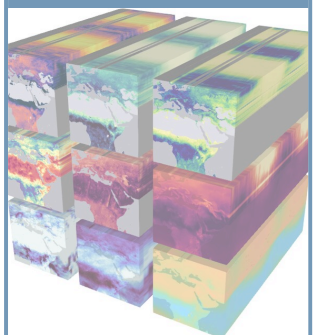
Principal components are sorted by the **variance** they explain: PC 1 is the most dominant direction, PC 2 the next, …

For the correlated Sentinel-2 bands, **2 of 12 components** keep more than 90 % of the variance.

The stronger the correlation, the fewer components we need.

Lecture Overview

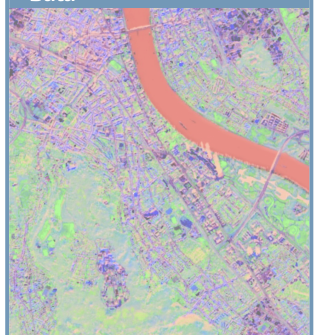
1 · More Data ✓



Curse of Dimensionality

→

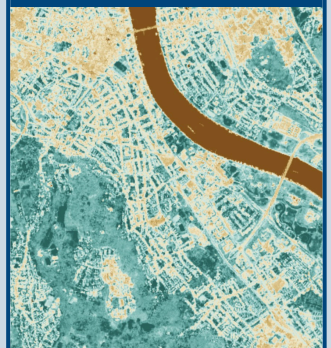
2 · Less Redundant Data ✓



Decorrelation through PCA

→

3 · More Descriptive Data



Domain Knowledge

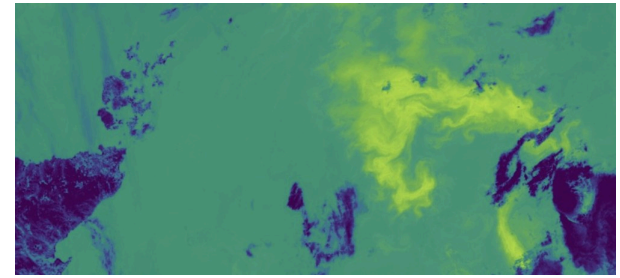
→

4 · Better (Non-linear) Models



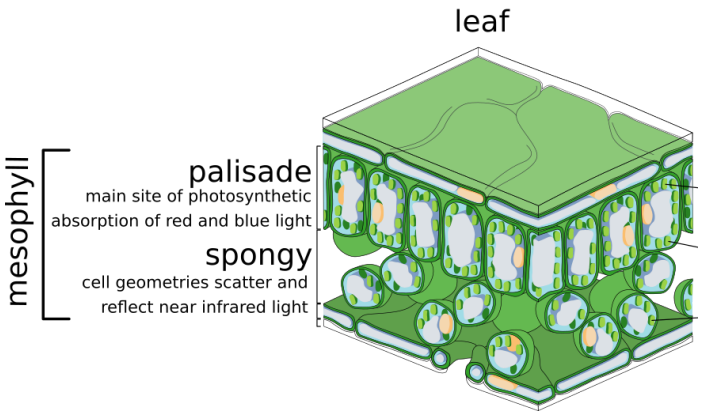
Random Forests

Choosing Better Features through Domain Knowledge

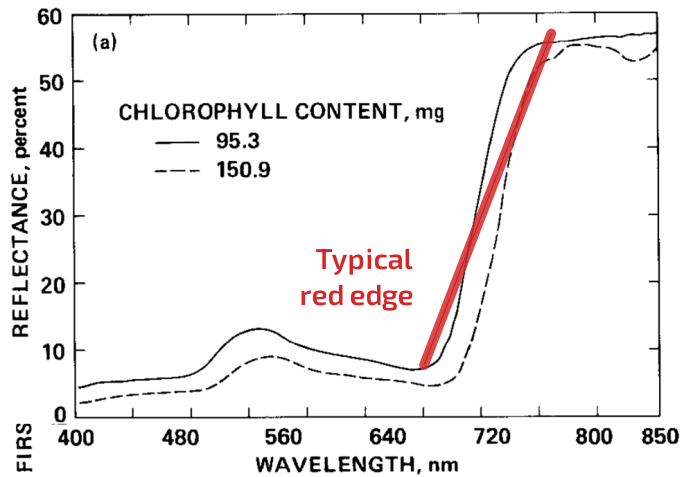


Chlorophyll-a index from Sentinel-3 OLCI, algal bloom off Scotland (2013-06-14).

Recap: Vegetation Spectra



1. **absorption** of blue (≈ 450 nm) and red (≈ 670 nm) light by chlorophyll a/b
2. strong **reflectance** of near-infrared light (> 750 nm) due to scattering in the leaf structure

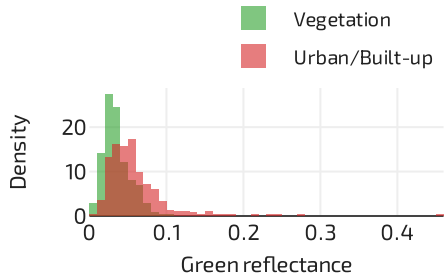


Curran, P. J., Dungan, J. L., & Gholz, H. L. (1990). Exploring the relationship between reflectance red edge and chlorophyll content in slash pine. *Tree Physiology*, 7(1-4), 33–48.

A Better Feature: NDVI

Raw bands

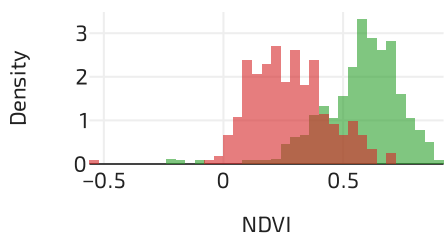
Pt	Red	Green	Blue	NIR	Land Cover
Pt1	0.08	0.11	0.09	0.42	**Vegetation**
Pt2	0.21	0.19	0.15	0.05	**Built-up**
⋮	⋮	⋮	⋮	⋮	⋮



Engineered feature

$$\text{NDVI} = \frac{\text{NIR} - \text{Red}}{\text{NIR} + \text{Red}}$$

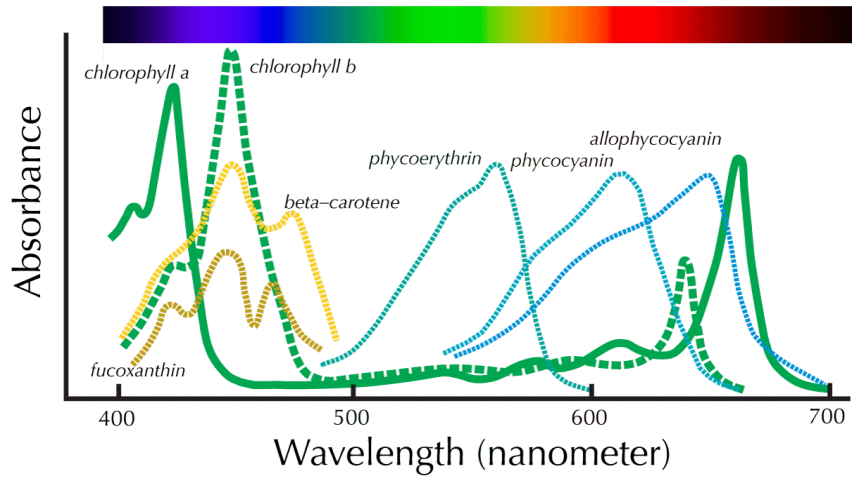
Pt	NDVI	Land Cover
Pt1	0.68	**Vegetation**
Pt2	-0.62	**Built-up**
⋮	⋮	⋮



Photosynthetically Active Pigments

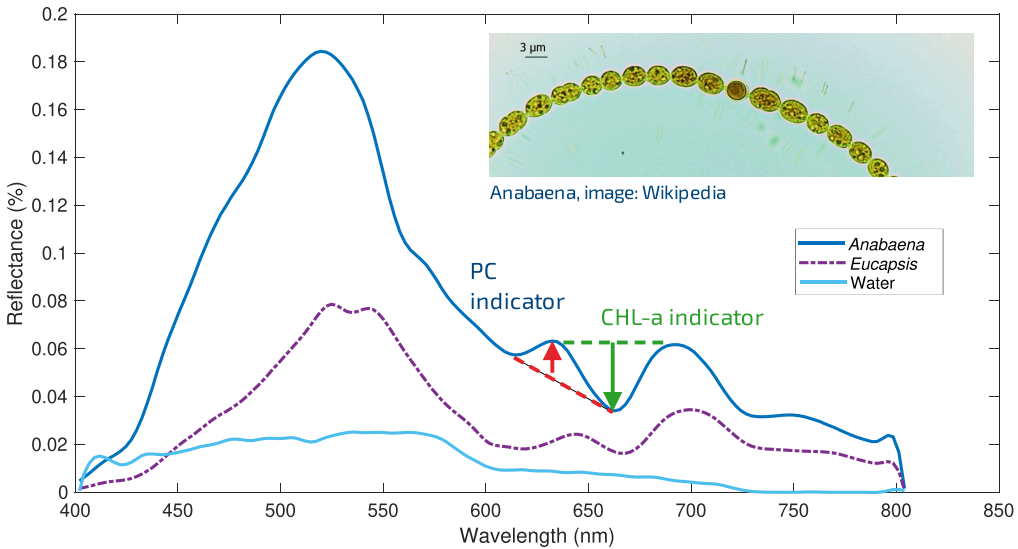
- **chlorophylls** and other pigments capture the energy of light
- different organisms use **different pigments** for photosynthesis
- terrestrial plants use both **CHL-a and CHL-b**
- algae and cyanobacteria do not: e.g. **phycocyanin**, phycoerythrin, fucoxanthin

Each pigment leaves its own fingerprint in the spectrum.



Source: York University, Photosynthesis lab manual

Indicators for CHL-a and Phycocyanin (PC)



Modified from: Legleiter, C. J., et al. (2022). Spectral mixture analysis for surveillance of harmful algal blooms (SMASH). *Remote Sensing of Environment*, 279, 113089.

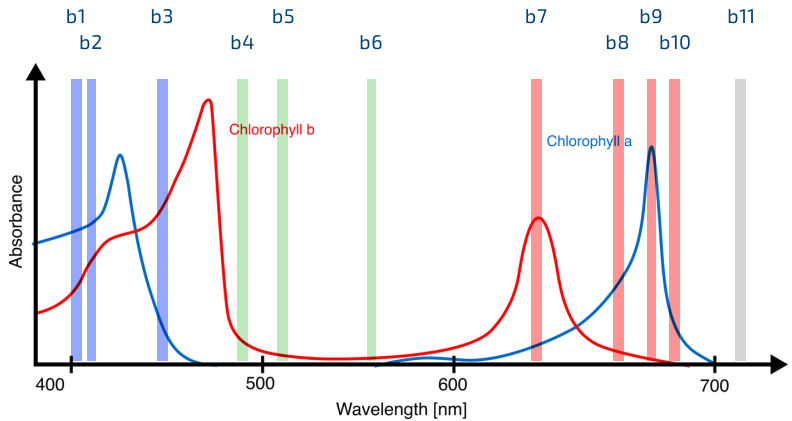


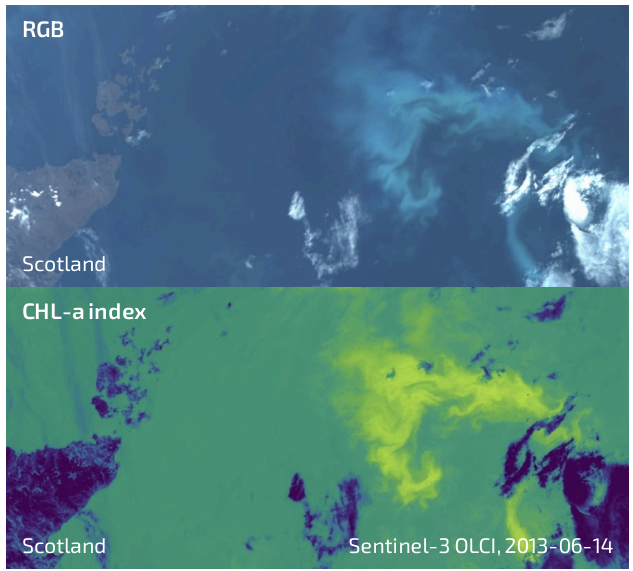
Eucapsis, image: Jason Oyadomari, Michigan Technological University, USA

The **shape** of the spectrum around 620–700 nm distinguishes pigments.

Example: Design a CHL-a Index

Sentinel-3 OLCI bands overlaid with chlorophyll absorbance





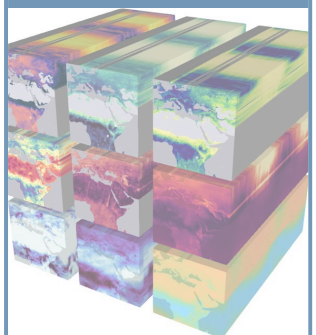
Pick two OLCI bands and combine them like NDVI, as a **normalized difference**:

$$\text{index} = \frac{b_i - b_j}{b_i + b_j}$$

Exercise script (Google Earth Engine) · Solution

Lecture Overview

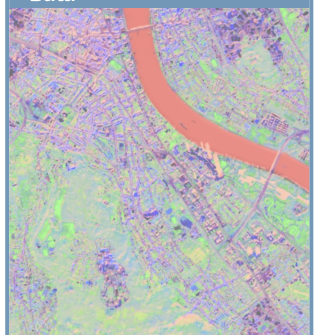
1 · More Data ✓



Curse of Dimensionality

→

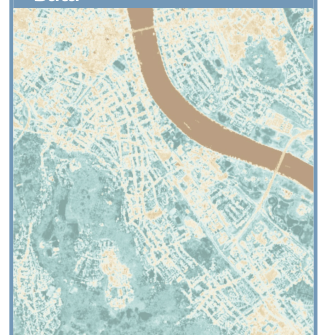
2 · Less Redundant Data ✓



Decorrelation through PCA

→

3 · More Descriptive Data ✓



Domain Knowledge

→

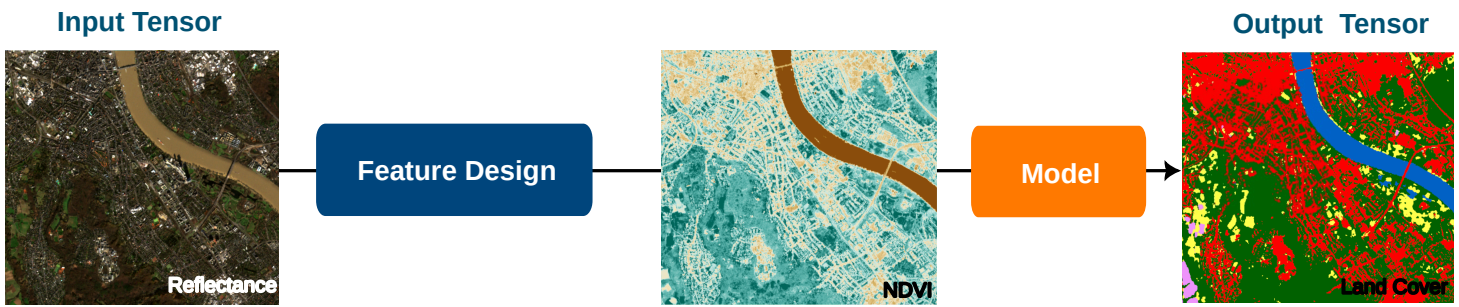
4 · Better (Non-linear) Models



Random Forests

Machine Learning Pipeline

Good, hand-selected features combined with a suitable model determine the quality of the result.



● This lecture · More Data, PCA & Better Features ● This lecture · Decision Trees & Random Forests

Non-linear models: Decision Trees & Random Forests



Image by Bing Image Generator

Decision Trees: The Core Idea

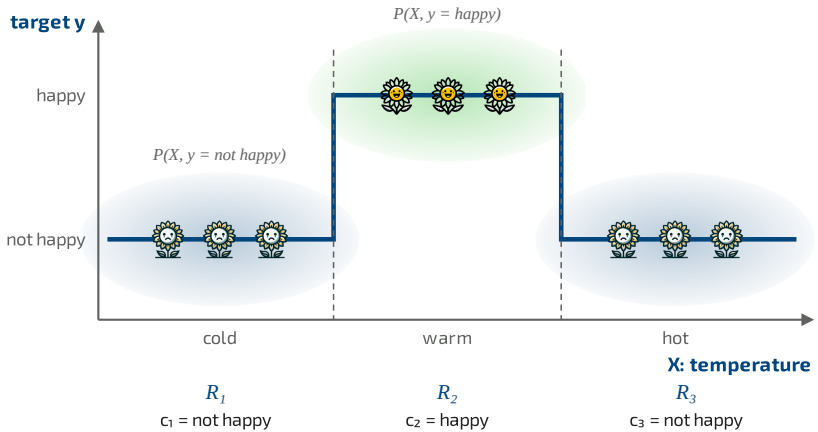
We categorize data through a **chain of simple decisions**.

- each decision is a simple test on a single feature (e.g., *is the temperature below 10 °C?*)
- implemented in computer science as a **binary tree**: a graph data structure in which each node has at most two children
- a universal tool for **classification and regression** of non-linear relationships



Image by Bing Image Generator

Linear Models Fail in Non-linear Situations



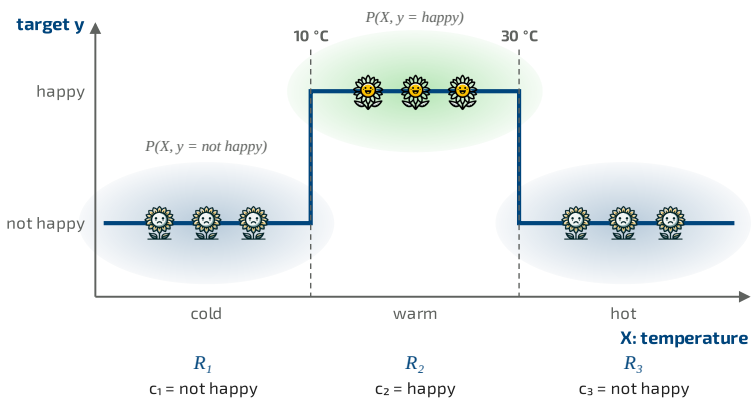
We need a different model: a **piecewise constant** function over regions R_m

$$f(X) = \sum_{m=1}^{M} c_m \cdot 1_{(X \in R_m)}$$

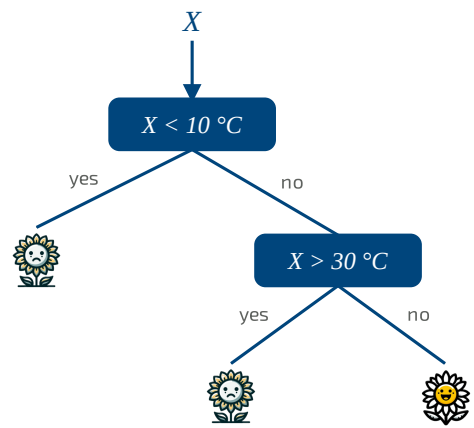
with one value c_m per region.

Decision Trees: Regions from Binary Decisions

$f(X) = \sum_{m=1}^{M} c_m \cdot 1_{(X \in R_m)}$ — regions defined by intervals: $R_1 = \{X \leq 10\,^\circ\mathrm{C}\}, R_2 = \{10\,^\circ\mathrm{C} < X \leq 30\,^\circ\mathrm{C}\}, R_3 = \{X > 30\,^\circ\mathrm{C}\}$

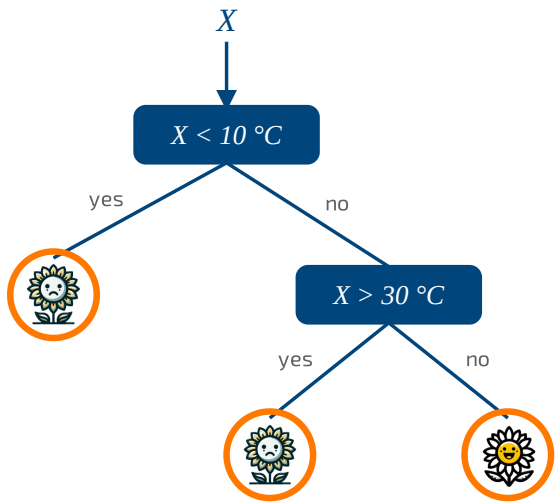


A simple decision tree



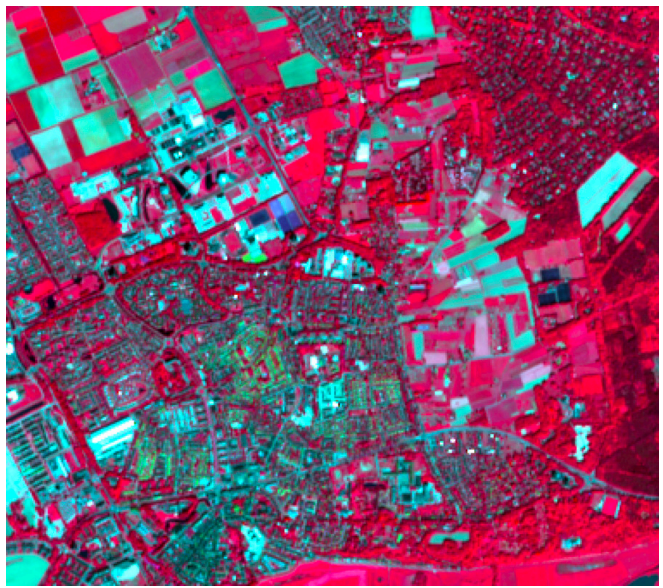
(drawn upside down: the root is at the top)

A Classification Decision Tree: Terminology

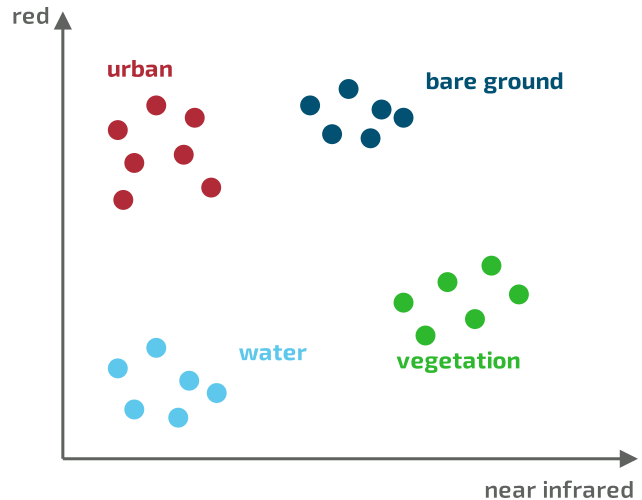


- **input features**: the data to be classified
- the **root node** starts the decision process
- **decision nodes** (split nodes) implement a **test function with discrete outcomes**; each test splits the input space into regions
- **leaf nodes** mark the end of a sequence of decisions; a single (output) class is associated with each leaf

Multivariate Data Example: Land Cover Classification

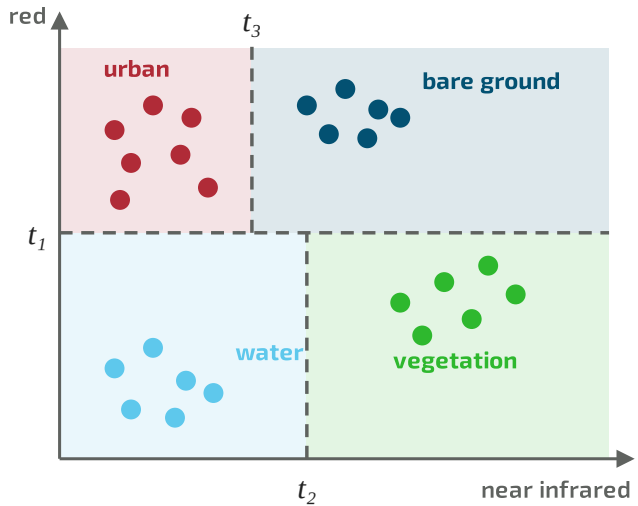


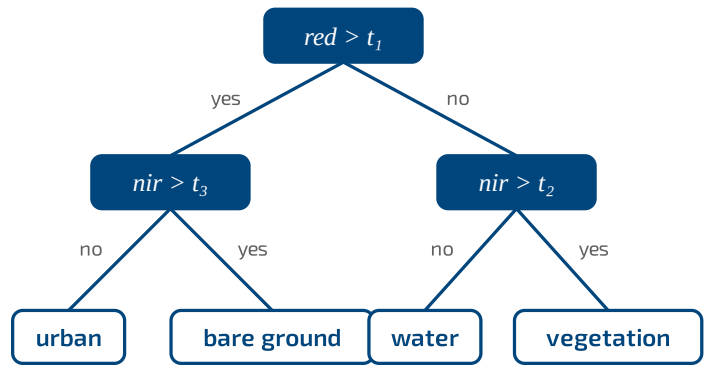
Sentinel-2, false-colour composite (Wageningen, NL). Contains modified Copernicus Sentinel data.



Each pixel is a point in feature space: its **red** and **near-infrared** reflectance.

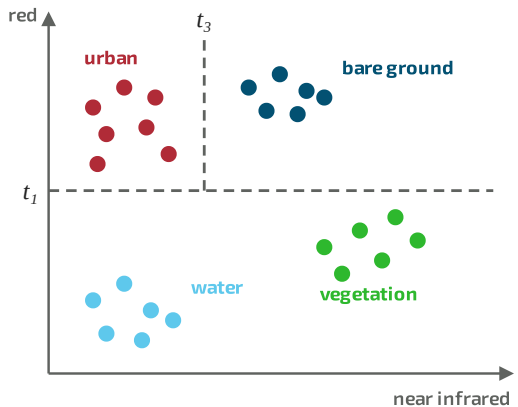
Decision Tree for Multivariate Data

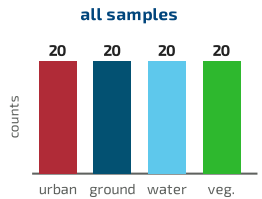




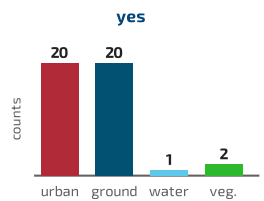
Every split is **axis-aligned**: each decision tests one feature against one threshold, so the regions are rectangles.

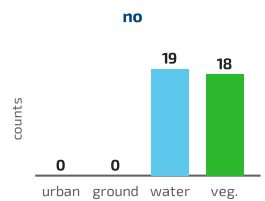
Increasing Node Purity: Split & Repeat



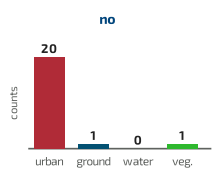


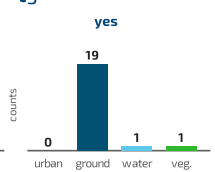
$red > t_1$





$nir > t_3$





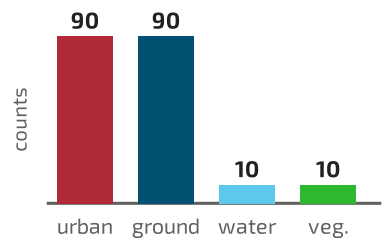
Idea: every decision should increase the node purity.

… and so on

Quantifying Node Purity: Entropy

$$H(\mathbf{p}) = -\sum_i p_i \log(p_i) \qquad (\text{here with } \log_{10})$$

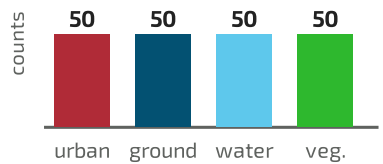
lower entropy: more pure



$p_1 = p_2 = \frac{90}{200} = 0.45, \quad p_3 = p_4 = \frac{10}{200} = 0.05$

$H = -(2 \cdot 0.45 \log 0.45 + 2 \cdot 0.05 \log 0.05) \approx \mathbf{0.44}$

higher entropy: less pure



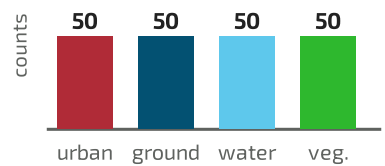
$p_{1,2,3,4} = \frac{50}{200} = 0.25$

$H = -4 \cdot 0.25 \log 0.25 \approx \mathbf{0.60}$

Pen & Paper Exercise: Gini Index

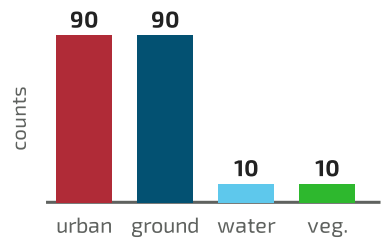
$$G(\mathbf{p}) = \sum_i p_i(1 - p_i)$$

higher Gini index: less pure



$G = 4 \cdot 0.25 \cdot 0.75 = \mathbf{0.75}$

lower Gini index: more pure



$G = 2 \cdot 0.45 \cdot 0.55 + 2 \cdot 0.05 \cdot 0.95 = \mathbf{0.59}$

Exercise: calculate the Gini index for both distributions.

Classification versus Regression Trees

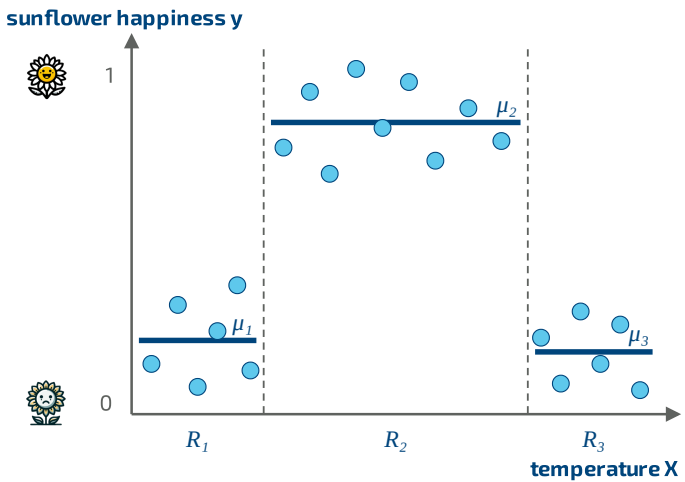
So far, we only considered **classification trees**: each leaf represents one class (the majority of the training samples in the leaf).

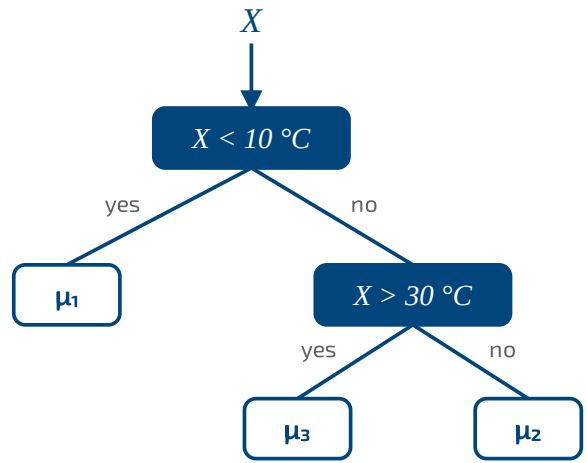
- in **classification** problems, the target variable is **categorical** (e.g., happy / not happy sunflower)
- in **regression** problems, the target variable is **continuous** (e.g., a happiness score between 0 and 1)

Regression trees follow exactly the same principles, but use

- a different metric for node purity: the **sample variance**
- a different interpretation of the leaves: the **sample mean**

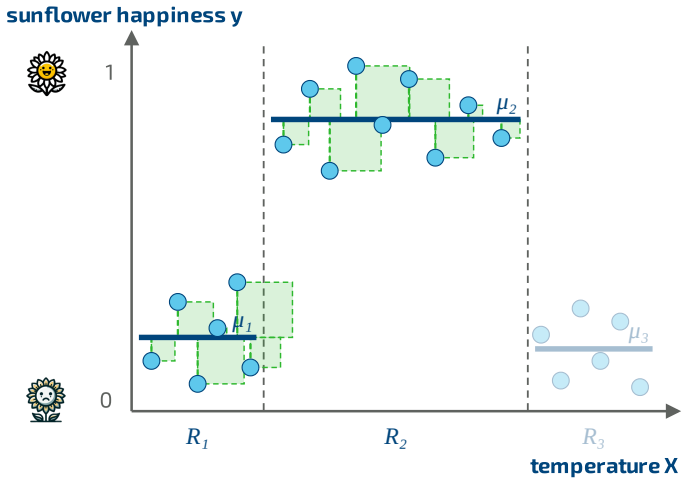
Regression Trees: Leaves Return the Average





$$\mu_m = \frac{1}{N_m} \sum_{i:\, x_i \in R_m} y_i$$

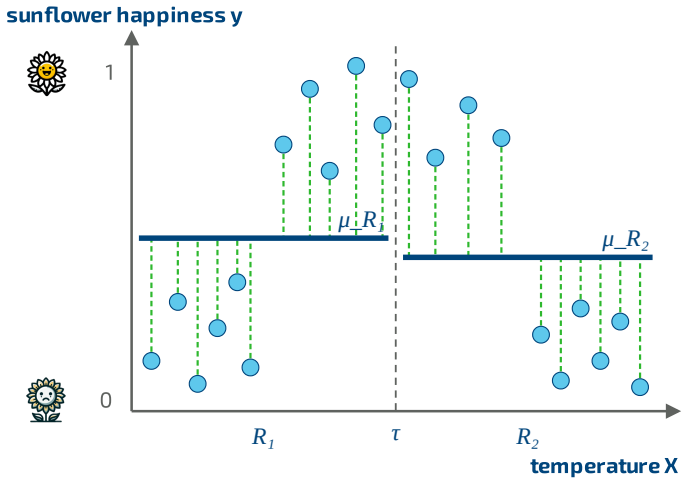
Sample Variance as Node Purity



$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \mu)^2$$

After splitting at $10\ ^\circ\mathrm{C}$: smaller average squares = lower variance = higher node purity.

How to Find the Threshold τ?



Formal definition of the regions by the threshold

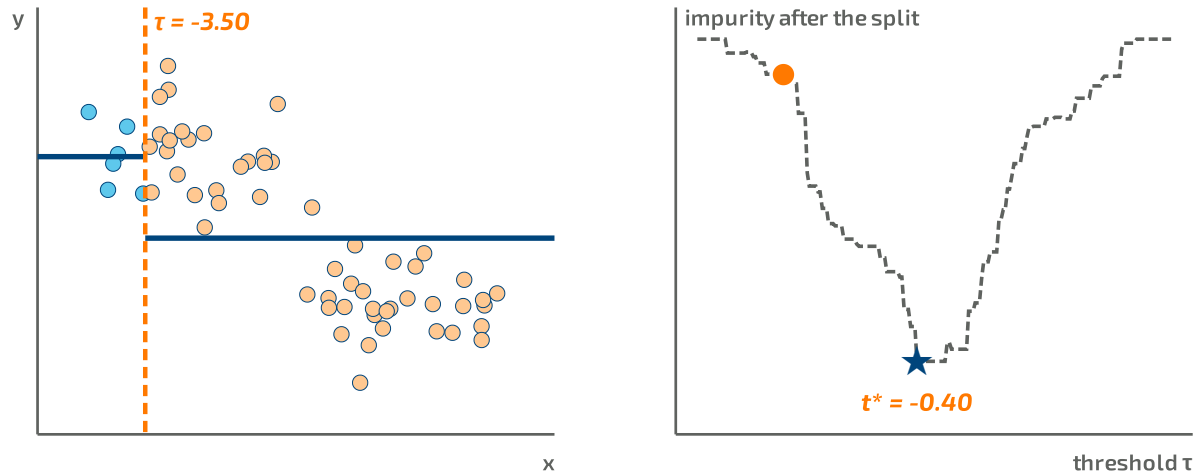
$$R_1 = \{X \mid X < \tau\}, \qquad R_2 = \{X \mid X \geq \tau\}$$

Minimization objective

$$t^* = \arg\min_{\tau} \underbrace{\sum_{i:\, x_i \in R_1} (y_i - \mu_{R_1})^2}_{\text{left variance}} + \underbrace{\sum_{i:\, x_i \in R_2} (y_i - \mu_{R_2})^2}_{\text{right variance}}$$

The impurity is a function of τ — we look for its minimum t^*.

Naïve Approach: Try Different Thresholds

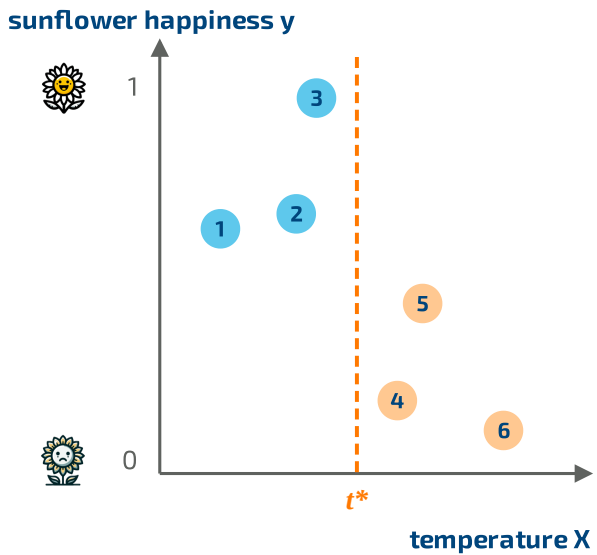


τ impurity 5.61

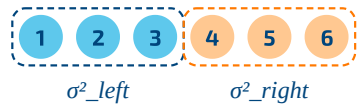
Easy to implement — but we need to choose which thresholds to test. Drag the slider to scan the thresholds; the best recorded threshold is t^*.

Original Colab notebook: threshold search

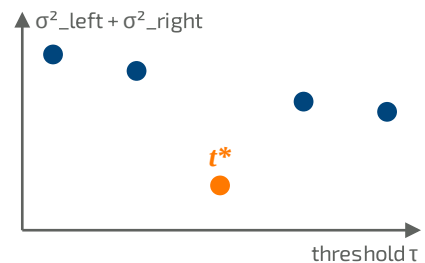
Binary Splitting



1. sort data points by X (index 1–6)



2. for every pair of neighbours i, i+1:
   try τ between them, record $\sigma^2_{left} + \sigma^2_{right}$



Only thresholds between neighbouring points change the split — we test as many thresholds as we have points.

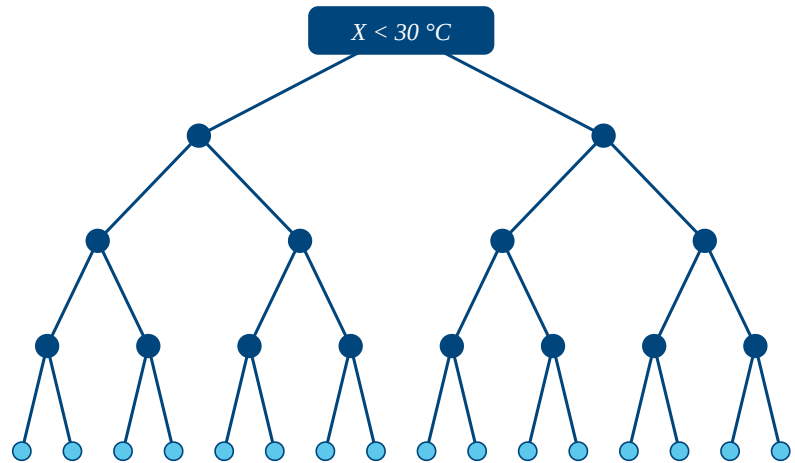
Growing a Tree: Recursive Binary Splitting

Growing a decision tree:

- perform binary splitting
- recursively for each decision node
- to maximize node purity

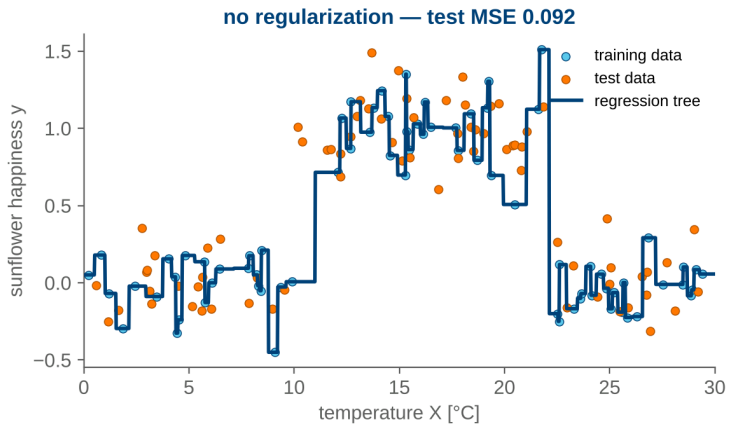
When to stop splitting?

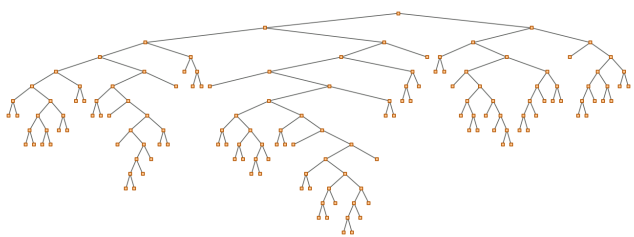
Without a stopping rule we end up with one sample per leaf = a 1-nearest-neighbour regressor / classifier.



Overfitting of Decision Trees

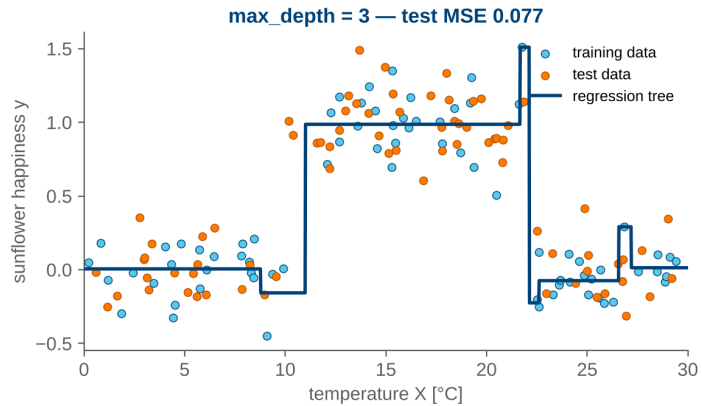
- the tree fits the training data perfectly (zero variance in every leaf)
- the tree is extremely large and uninterpretable
- test MSE: **0.092** happiness

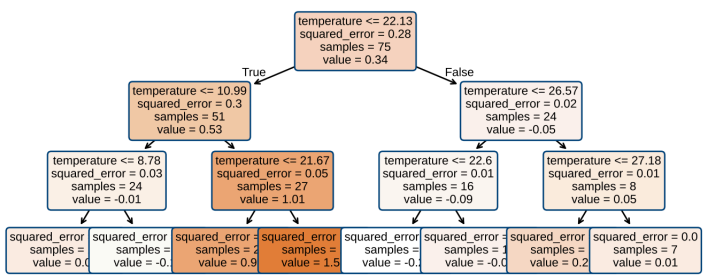




Regularization: Restricting the Depth

- we can manually restrict the depth
- the tree remains interpretable
- test MSE lower: **0.077** happiness





Original Colab notebook: [regularization of decision trees]

Regularization: Tree Pruning

Two families of algorithms restrict the tree complexity and counteract overfitting.

Slow growers (pre-pruning)

Strategy: grow the tree slowly, with additional checks during growing.

- simple checks: **maximum depth**, **maximum number of leaf nodes**
- add a decision node only if the **increase in node purity** is above a threshold

Fast cutters (post-pruning)

Strategy: grow the full tree, then remove unimportant nodes.

- **cost complexity pruning**: we identify the weakest link in the tree and remove it

Recap: Decision Trees

Advantages

- **inexpensive** to construct
- extremely **fast at prediction** time
- **easy to interpret** for small trees
- **robust to noise** (when regularized)
- can easily handle **redundant or irrelevant attributes** (unless the attributes are interacting)

Disadvantages

- decision trees are prone to **overfitting** and need regularization
- **greedy** approaches often fail to find the best tree (restricting growth is too short-sighted)
- do **not take interactions between features** into account: each decision boundary involves only a single attribute

Bagging & Random Forests

Increasing robustness by redundancy



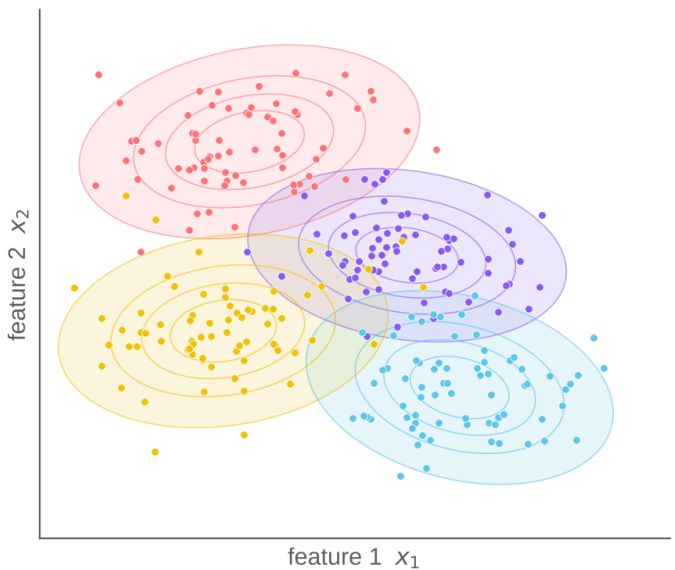
Bagging



Random Forests

Images by Bing Image Generator (DALL·E 3): "a stylized image of a bag of green trees", "... of a random forest", white background.

An Artificial Classification Dataset



Assume four distributions, one per class, over a random feature vector $\mathbf{X} = (x_1, x_2)$:

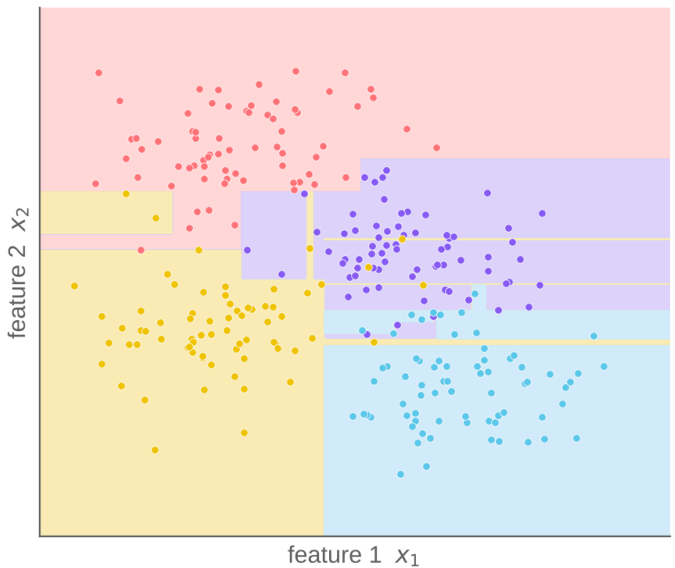
- class 1: $p(x_1, x_2 \mid y = 1)$
- class 2: $p(x_1, x_2 \mid y = 2)$
- class 3: $p(x_1, x_2 \mid y = 3)$
- class 4: $p(x_1, x_2 \mid y = 4)$

We **sample a dataset** from the distribution

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} \sim p(\mathbf{X}, Y)$$

Remember: we are always working with distributions.

Let's Fit a Decision Tree to the Data

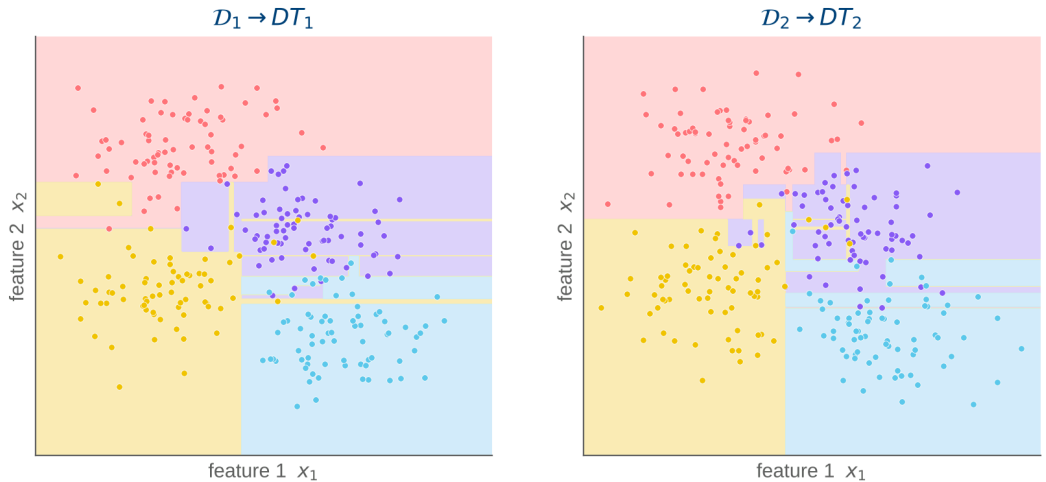


Again, we split the feature space into **regions**.

The modes (i.e., centers) of the data distributions are well assigned to the correct classes.

But the uncertain, overlapping region is not well defined: thin slivers and islands follow individual training points.

Different Samples Lead to Very Different Trees



Low bias: each tree fits its own training data perfectly.

High variance: the trees are very different on different dataset samples $\mathcal{D}_1, \mathcal{D}_2 \sim p(\mathbf{X}, Y)$.

Bias–Variance Trade-off

Finding a good balance between fitting and overfitting is the fundamental challenge in machine learning. For tree-like classifiers:

Strategy 1: reduce tree complexity

(recap: seen in Part 1)

- greedy: restrict the tree complexity while growing
- non-greedy: post-hoc pruning of the tree



"cut the tree"

Strategy 2: ensemble many trees

- bagging
- random forests



"grow a forest"

Let's Grow a Forest — Step 1: Bagging

$$\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} \sim p(\mathbf{X}, Y)$$

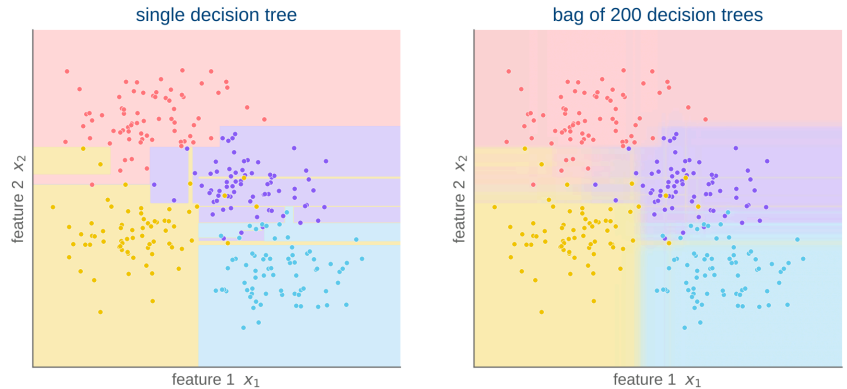
↓ randomly sampled subsets (with replacement), i.e., a "**bootstrap**" ↓

Bag of decision trees



$\mathcal{D}_1 \rightarrow DT_1$ $\mathcal{D}_2 \rightarrow DT_2$ $\mathcal{D}_3 \rightarrow DT_3$

each tree is deep and not pruned



Ensemble prediction:

- **majority voting** for classification
- **output averaging** for regression

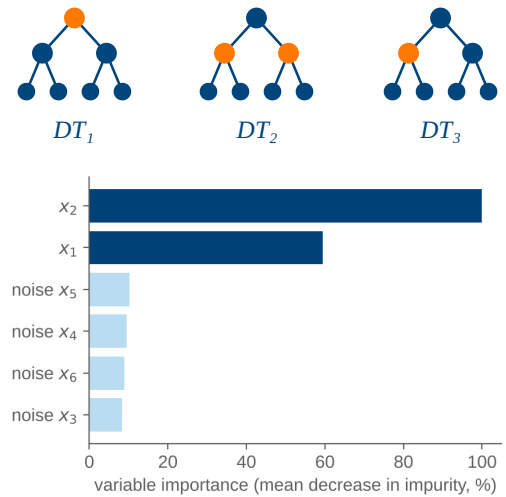
Effect: trees agree in the modes and disagree in the overlap — decision boundaries become blurry, a more realistic result.

Bagging Trees: Re-gaining Interpretability

- bagging **improves accuracy** over a single tree through increasing diversity in the decisions
- but we **lose interpretability**: each tree has different decision rules, and interpreting many rules in parallel is difficult

Variable importance — across all trees in the bag:

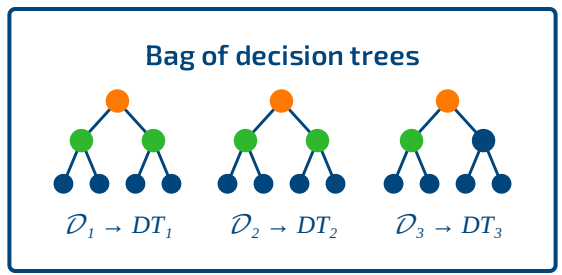
1. look at all decision nodes that involve the feature of interest ● (for instance, temperature)
2. measure the **increase in node purity** at each of these nodes and average it



Bagging Limitation: Correlated Trees

- bagging improves accuracy through increasing **diversity** in the decisions
- but bagged trees are still **highly correlated**: an important feature ● is split first, followed by the second most important feature ●, and so on…
- all trees in a bag look similar → this **correlates the trees**, reduces diversity & test accuracy

Idea: increase diversity by removing features from individual trees.



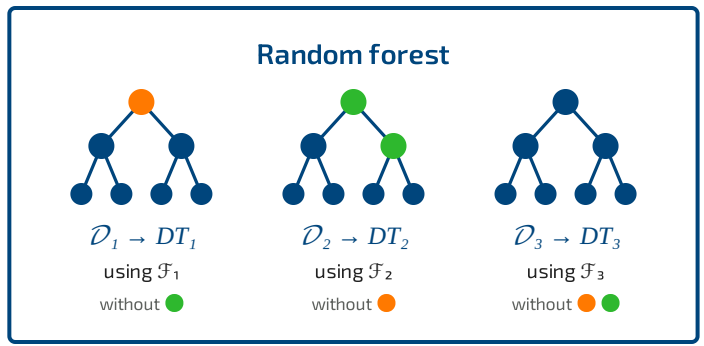
Random Forest — Step 2: Remove Features

Idea: increase diversity by removing features from individual trees.

Implementation: we take a random sample of predictors for each tree (usually $\sqrt{F}$ of F features, e.g., 4 out of 15).

In a random forest, each decision tree is

1. trained with a bootstrapped dataset $\mathcal{D}_i$
2. using a subset of features $\mathcal{F}_i$



Summary: Bagging and Random Forests

Bags & random forests

- each tree is trained on a different **bootstrapped** set of training data
- trees are trained separately from each other, **in parallel**



bag of decision trees

Random forests

- additionally, each tree is trained with a **different subset of features**



random forest

Recap: Decision Trees & Ensembles

Decision trees

- a simple chain of decisions
- well interpretable, but inaccurate on the test set (**low bias, high variance**)

Bag of trees

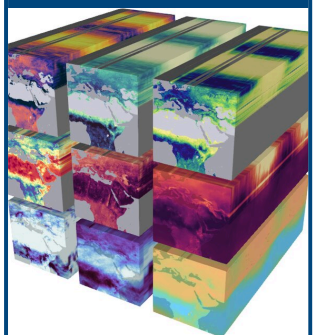
- an ensemble "bag" of trees, each trained on a different data subset ("bootstrap")
- better test accuracy through tree consensus; all trees learned **simultaneously**

Random forest

- like a bag, but each tree also uses a different **feature subset**
- trees are as **diverse** as possible; learned simultaneously

Lecture Overview

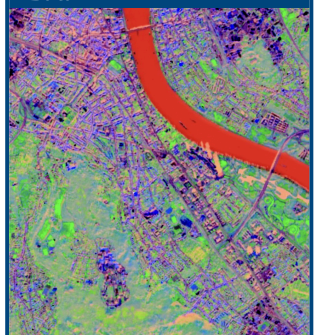
1 · More Data ✓



Curse of Dimensionality

→

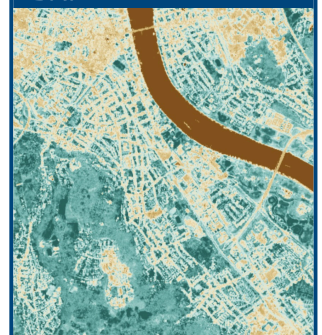
2 · Less Redundant Data ✓



Decorrelation through PCA

→

3 · More Descriptive Data ✓



Domain Knowledge

→

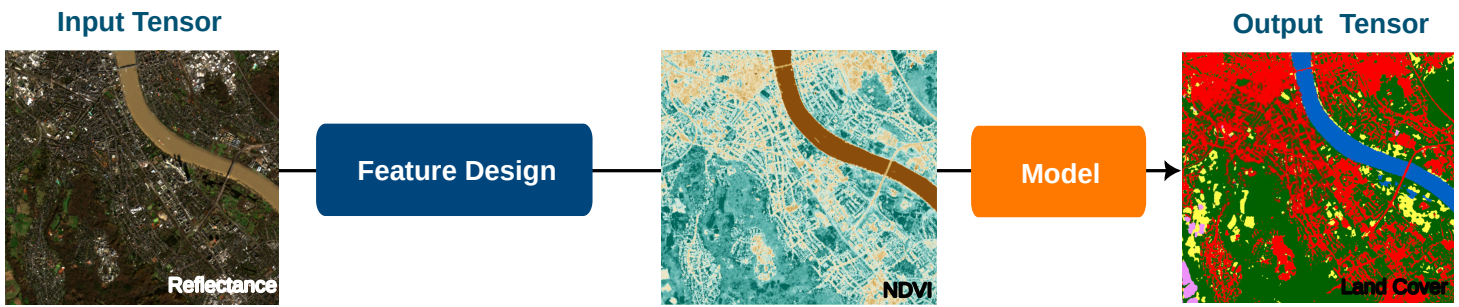
4 · Better (Non-linear) Models ✓



Random Forests

Machine Learning Pipeline

Good, hand-selected features combined with a suitable model determine the quality of the result.



● This lecture · More Data, PCA & Better Features ● This lecture · Decision Trees & Random Forests

Revisit the learning outcomes

Light gray fill: your group's before responses today · Click to respond again.

Understand the classic machine learning pipeline: feature design followed by a classification model.

Not comfortable — Very comfortable

Click to respond

Understand the curse of dimensionality and why we select features.

Not comfortable — Very comfortable

Click to respond

Understand how principal component analysis de-correlates data.

Not comfortable — Very comfortable

Click to respond

Know examples of problem-specific features designed with domain knowledge.

Not comfortable — Very comfortable

Click to respond

Understand the mechanisms behind decision trees and random forests.

Not comfortable — Very comfortable

Click to respond

Understand the principle of bias and variance in classification.

Not comfortable — Very comfortable

Click to respond