

There are 76 contributors for the current view.

Machine Learning for Digital Soil Mapping-Part 1

Ruhollah Taghizadeh
Soil Science and Geomorphology

E-mail: ruhollah.taghizadeh-mehrjardi@mnf.uni-tuebingen.de

Content

❑ Part 1

- Digital Soil Mapping
- Machine Learning
 - Decision Tree
 - Random Forest

❑ Part 2

- Lets practice!

SCORPAN model

$$S = f(s, c, o, r, p, a, n) + \varepsilon$$

S : Soil, at a specific point in space and time: soil classes, **Sc** or soil attributes, **Sa**

From Jenny's Equation

c : climate, climate properties of the environment;

o : organisms, vegetation;

r : topography, landscape attributes;

p : parent material, lithology;

a : age or time factor;

Additions:

s : soil, prior knowledge of the soil at a point;

n : space, relative spatial position;

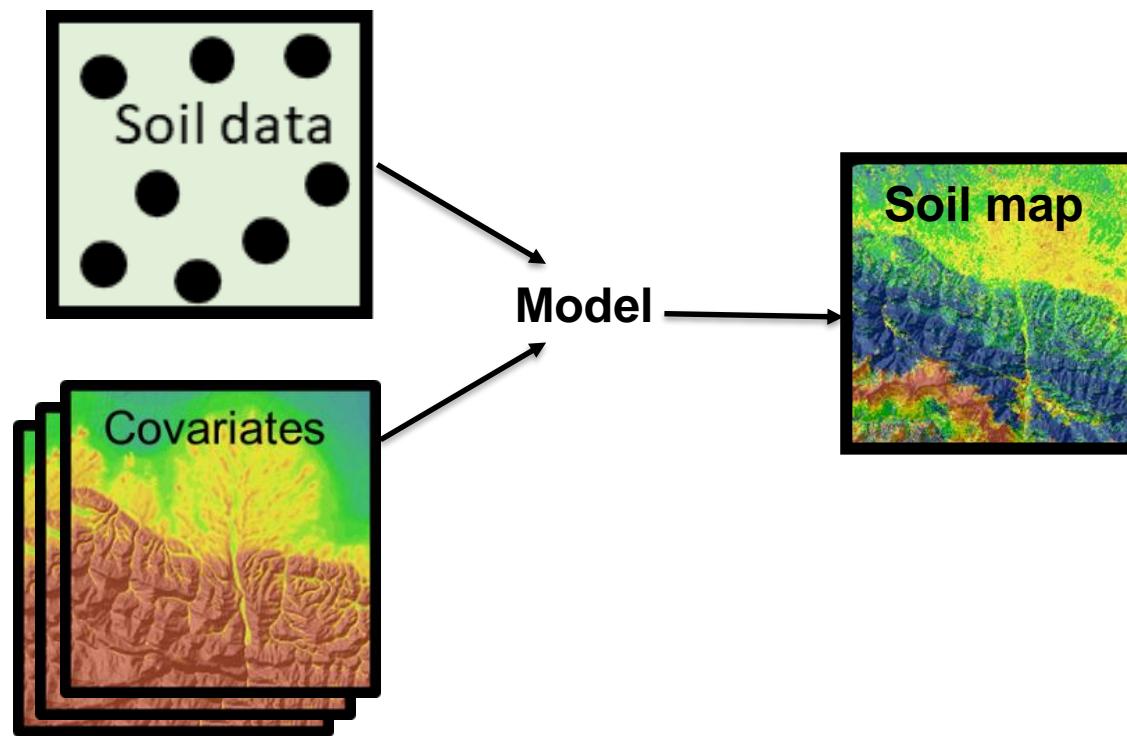
ε : auto-correlated random spatial variation.

f() : Quantitative function **f** linking **S** to **scorpan** factors

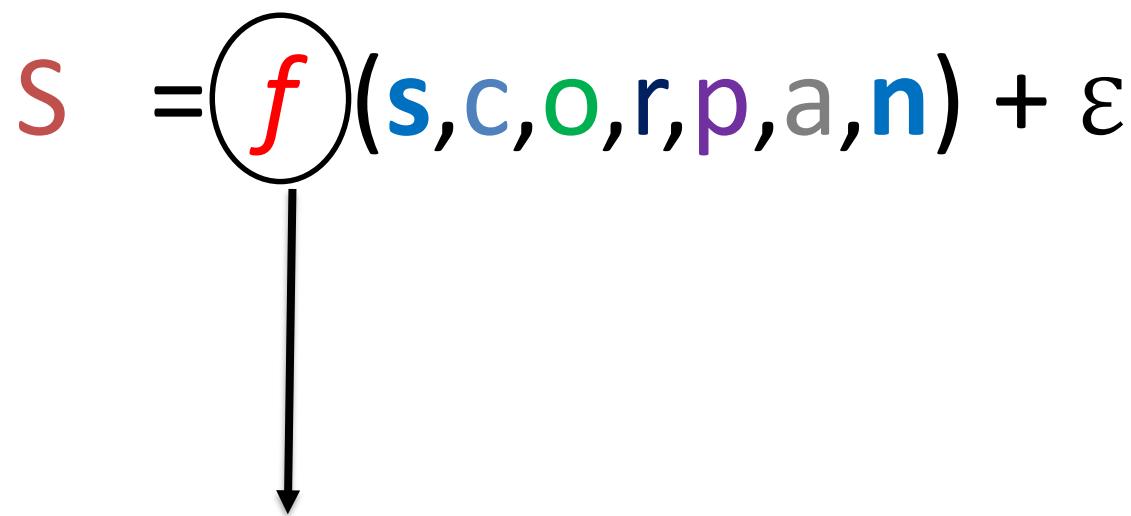
SCORPAN model

$$S = f(s, c, o, r, p, a, n) + \varepsilon$$

Soil data Model Covariates spatially dependent residuals



SCORPAN Model

$$S = f(s, c, o, r, p, a, n) + \varepsilon$$


Machine learning

f() : Quantitative function *f* linking *S* to *scorpan* factors

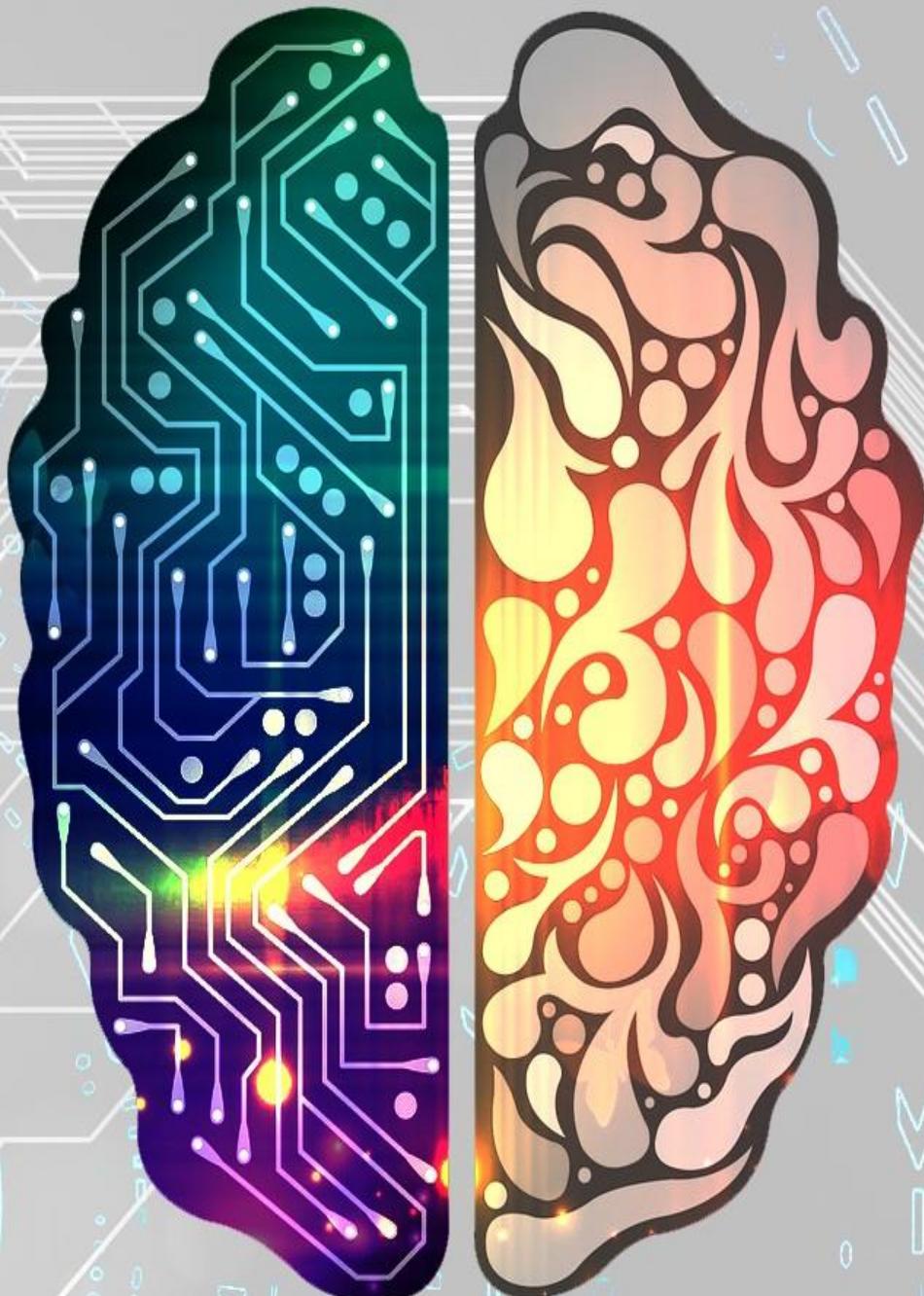
Sequence of DSM Steps



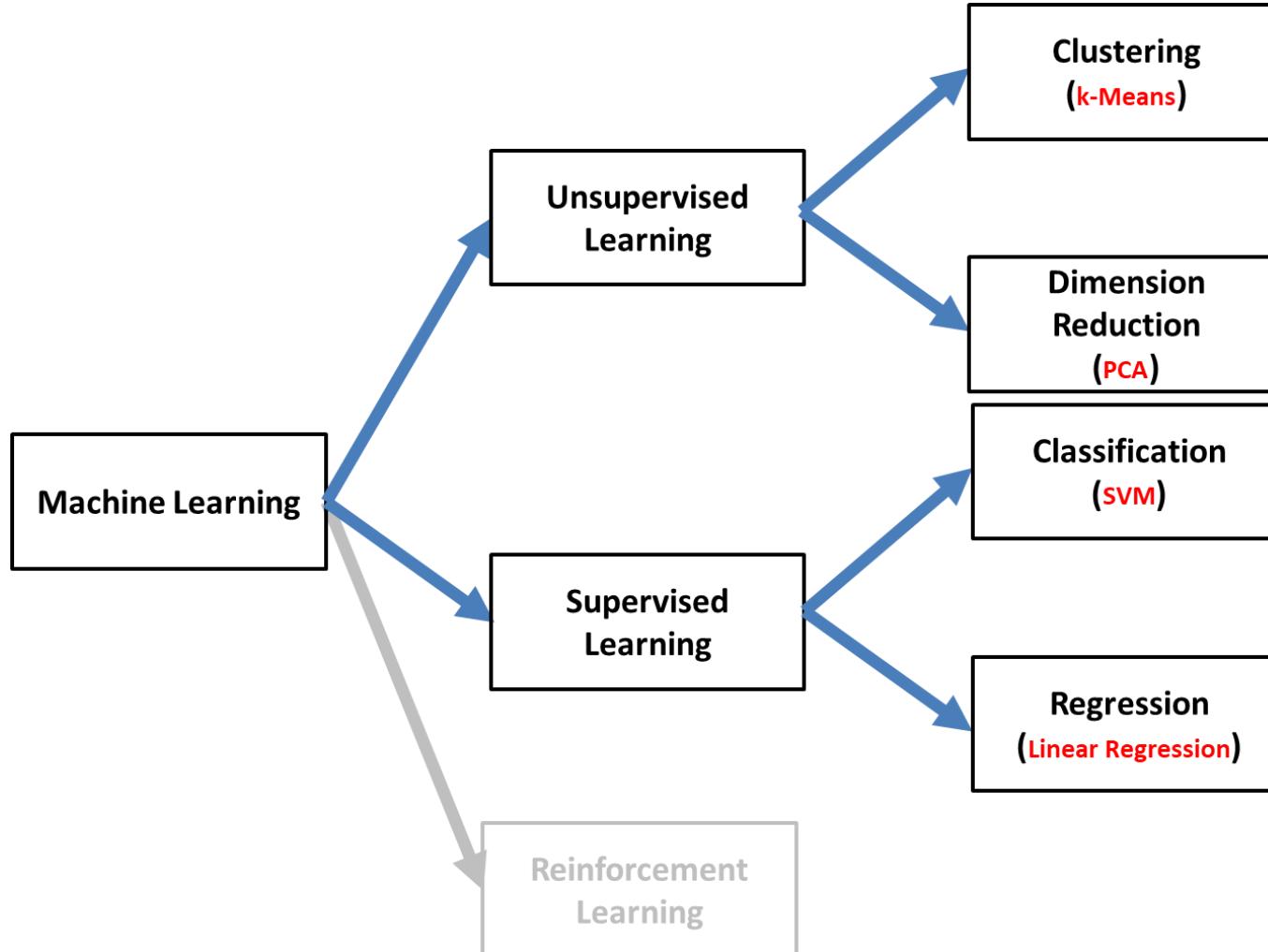
- 1 Environmental covariates, relevant as predictors of soil property/class, are derived from remote sensing, digital elevation, climatic datasets, ...
- 2 Soil samples are collected at the specified locations (e.g., Latin hypercube sampling) and soil property is measured in the laboratory.
- 3 Intersecting the covariates with the soil point observations.
- 4 Machine learning models (e.g., random forest) are trained using training data, and accuracy assessment is carried out using the test data set.
- 5 The ML models are applied to the entire study area in order to produce a soil property/class map.

Introduction: some terms

MACHINE LEARNING



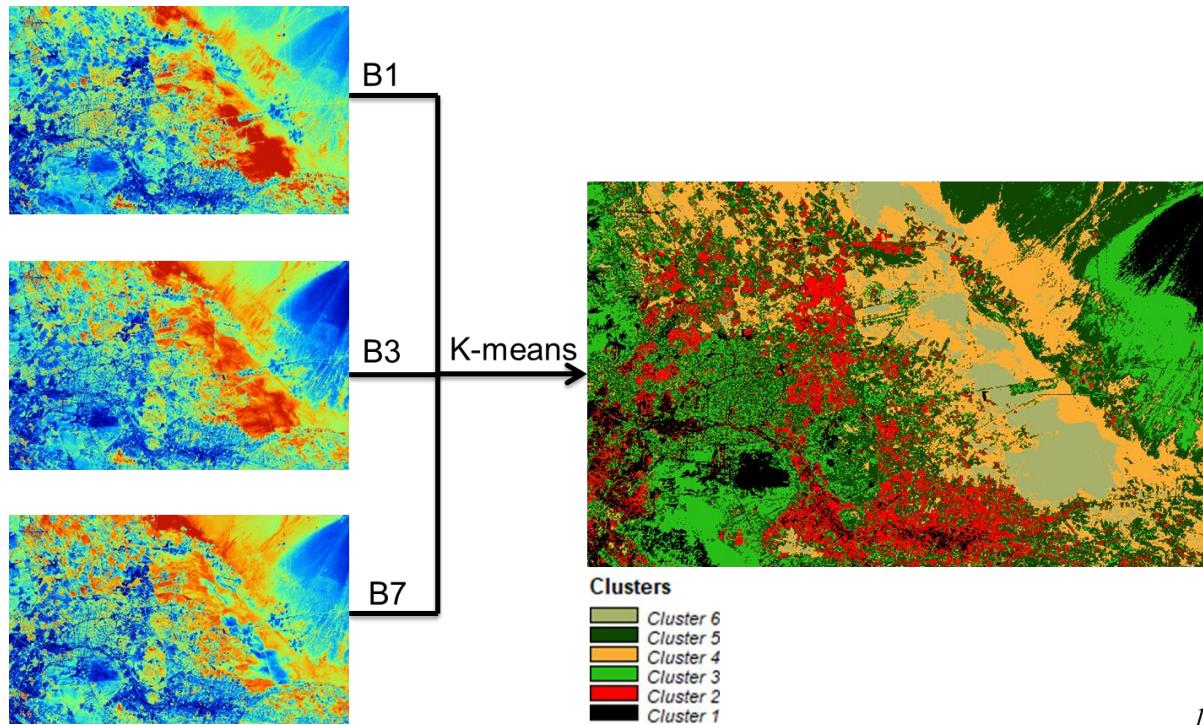
Types of Machine Learning



Unsupervised Learning

Unsupervised learning

- No response, just “**covariates**”, (x_i)
- Algorithm identifies clusters
- No training data required
- e.g., clustering of satellite image by similar values



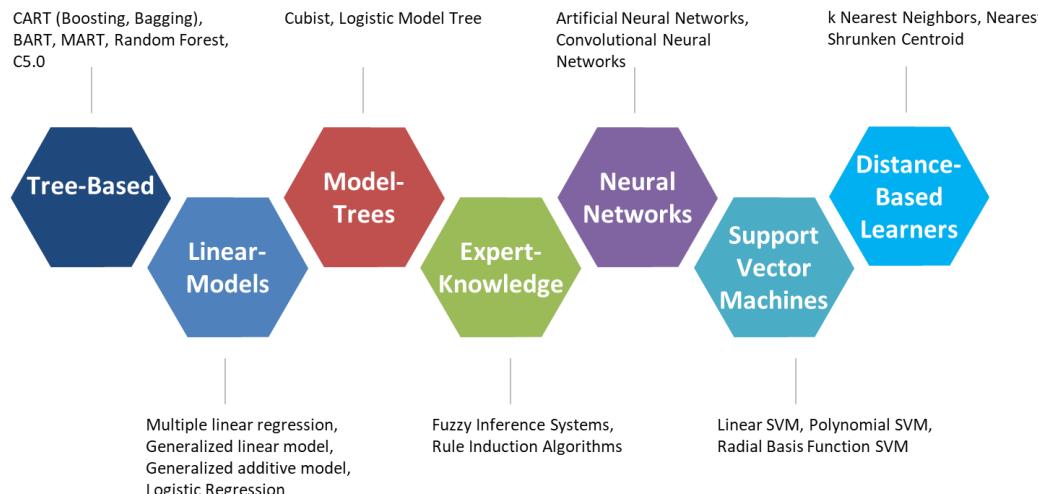
Supervised Learning

Supervised learning

- For each covariate value x_i there is also a response value y_i
- → e.g. random forest
- → **what we usually do for soil mapping**

Supervised learning

- – Regression: continuous responses, e.g. soil clay content, rainfall
- – Classification: categorical responses (binary or multinomial), e.g. soil type



Things to Keep in Mind

- Each model predicts **one realization** of a DSM
- **Effectiveness** of the model are dependent on:
 - Study Area (Extent and Resolution)
 - Target Variable
 - Environmental Covariates
 - Model Structure (e.g. Linear vs. Hierarchical)
- Each model has its own set of **hyperparameters** (i.e. model settings) that need to be optimized.
- Some models are more **computationally** efficient than others.
- Each model may produce drastically **different results** given the same inputs.
- Some models tend to **overfit**.

From Available Models

Decision
Tree

Random
Forest

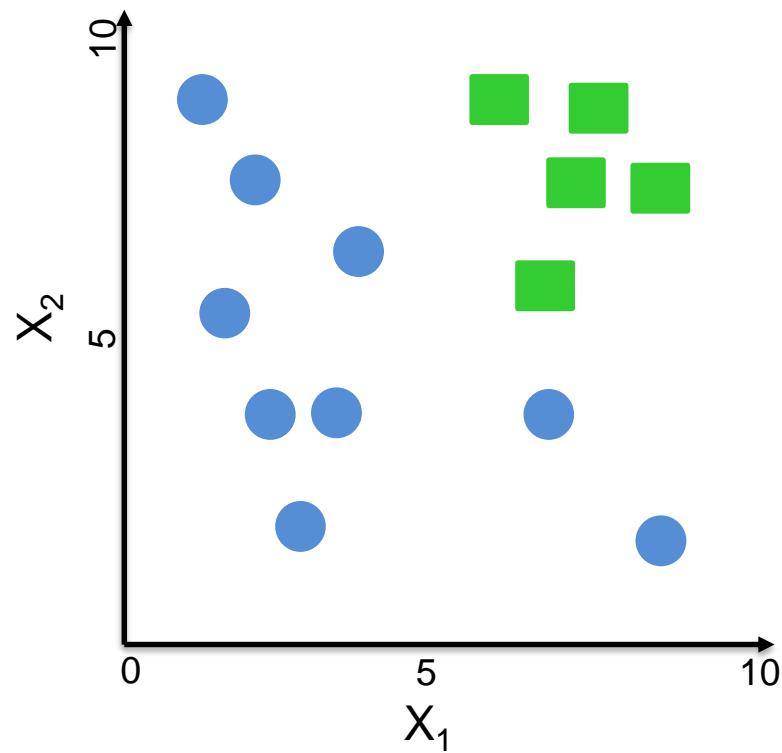
Decision Tree



Decision Tree: Definition

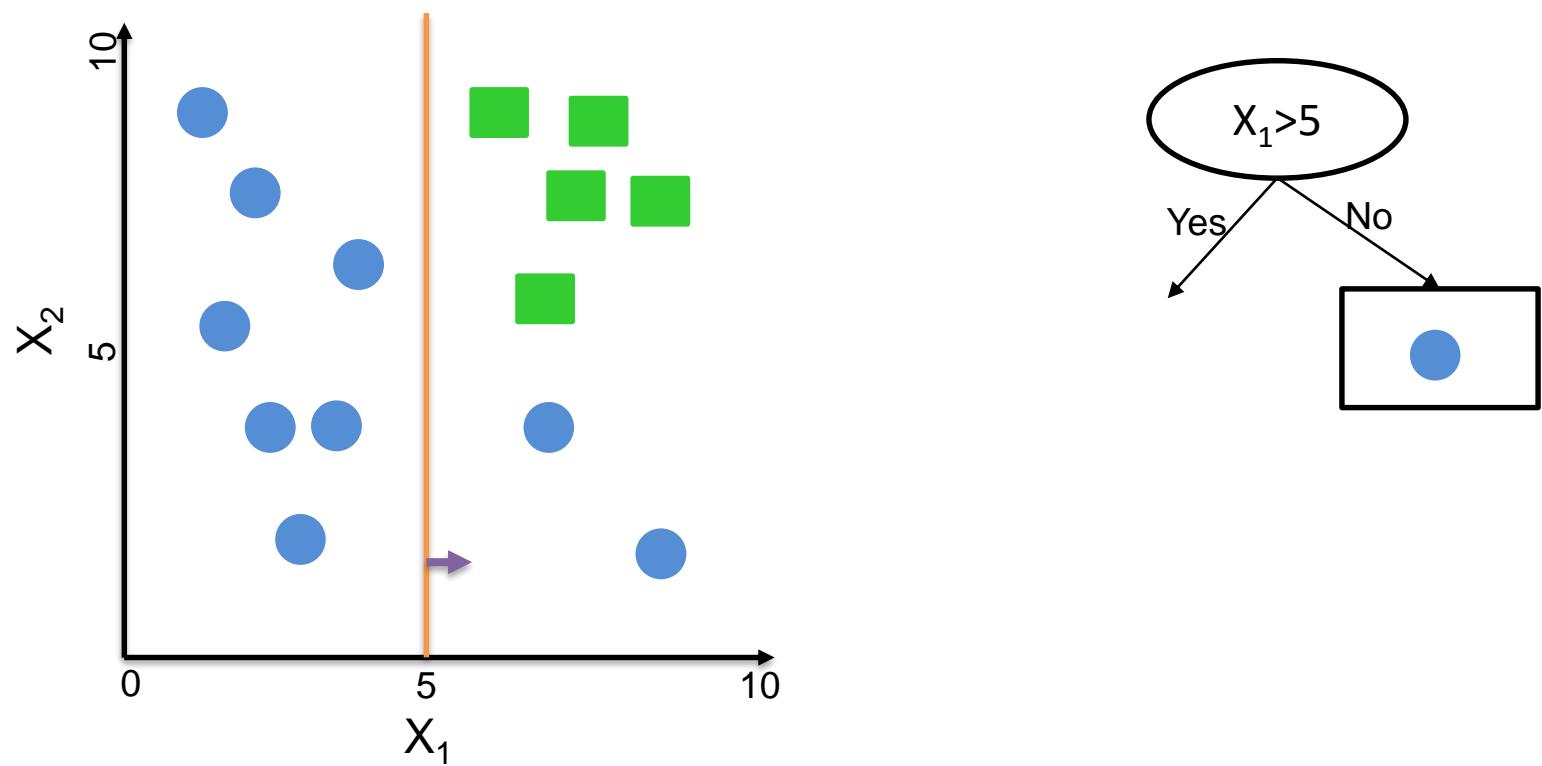
- Decision Tree is a supervised learning approach.
- It can be used for solving regression (e.g., clay content) and classification (e.g., soil types) problems.
- It partitions the data into subsets.
- The partitioning process starts with a binary split (Yes-No) and continues until no further splits can be made.
- It predicts the outcome by asking a set of **if-else** questions.
- It resembles a **tree-like graph, easy to understand**

Decision Tree: example I



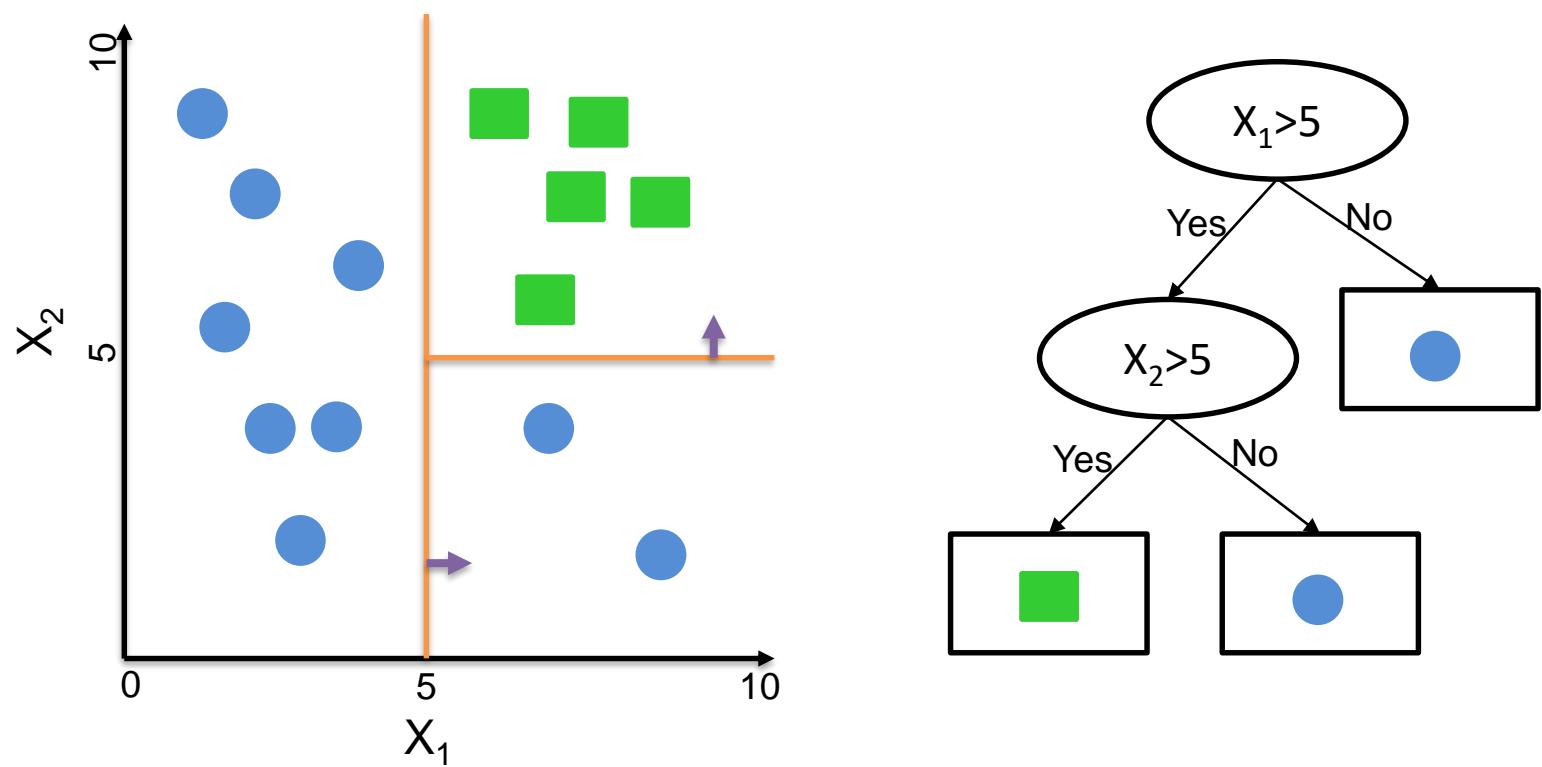
Example of a dataset and the corresponding decision tree. (Alpaydin, 2010)

Decision Tree: example I



Example of a dataset and the corresponding decision tree. (Alpaydin, 2010)

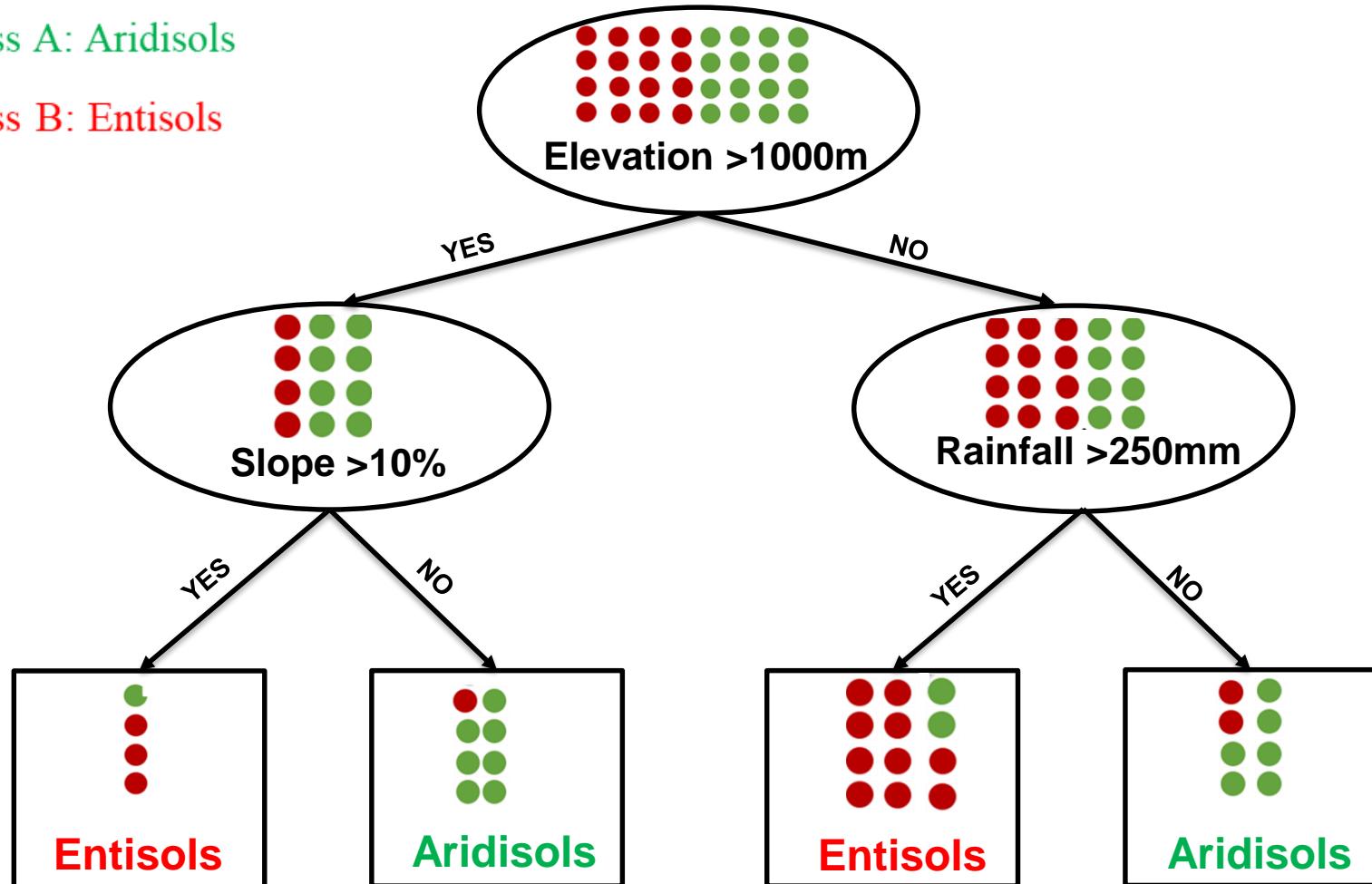
Decision Tree: example I



Example of a dataset and the corresponding decision tree. (Alpaydin, 2010)

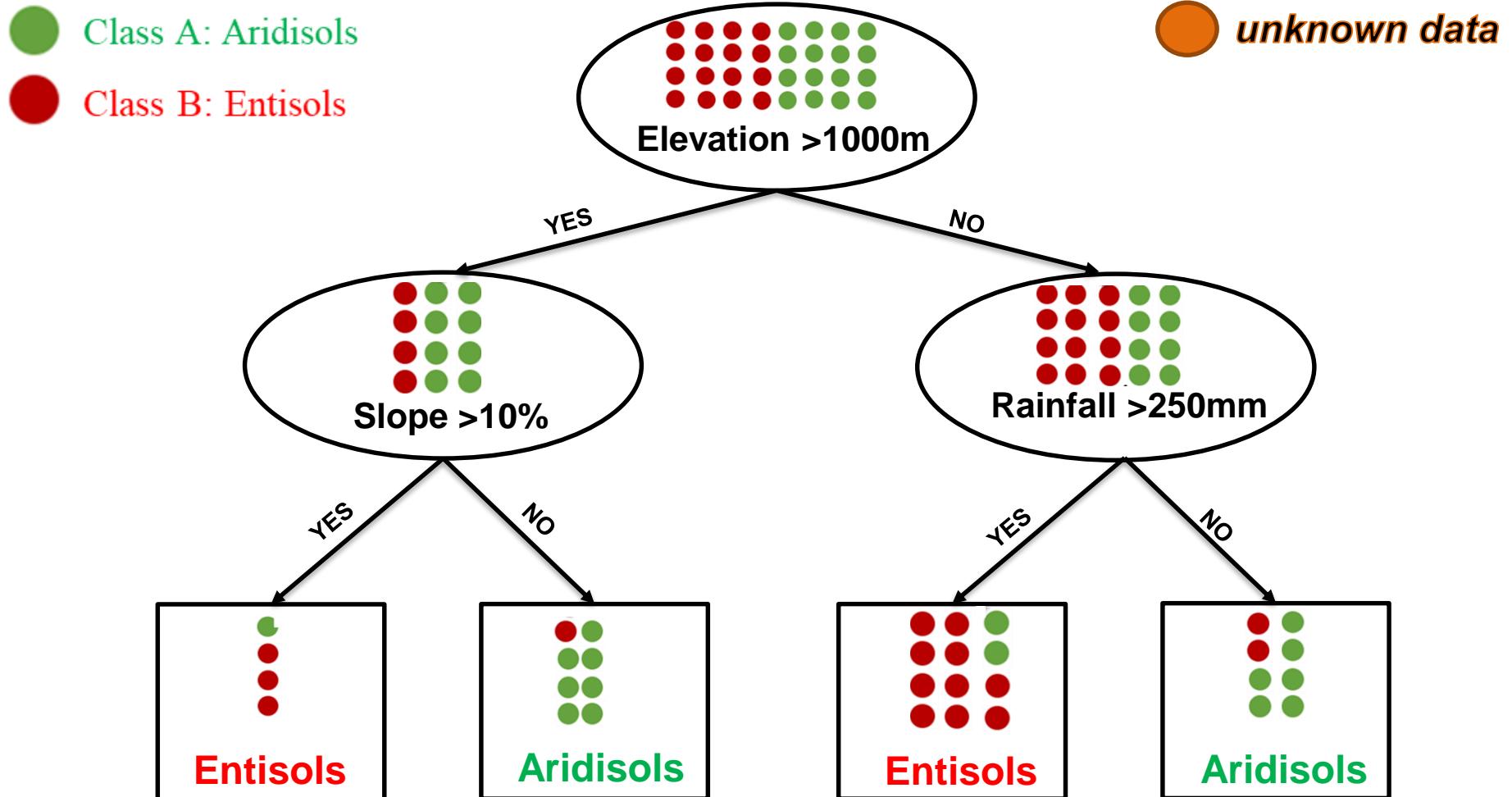
Decision Tree: example II

- Class A: Aridisols
- Class B: Entisols



A decision tree model to predict Entisols and Aridisols

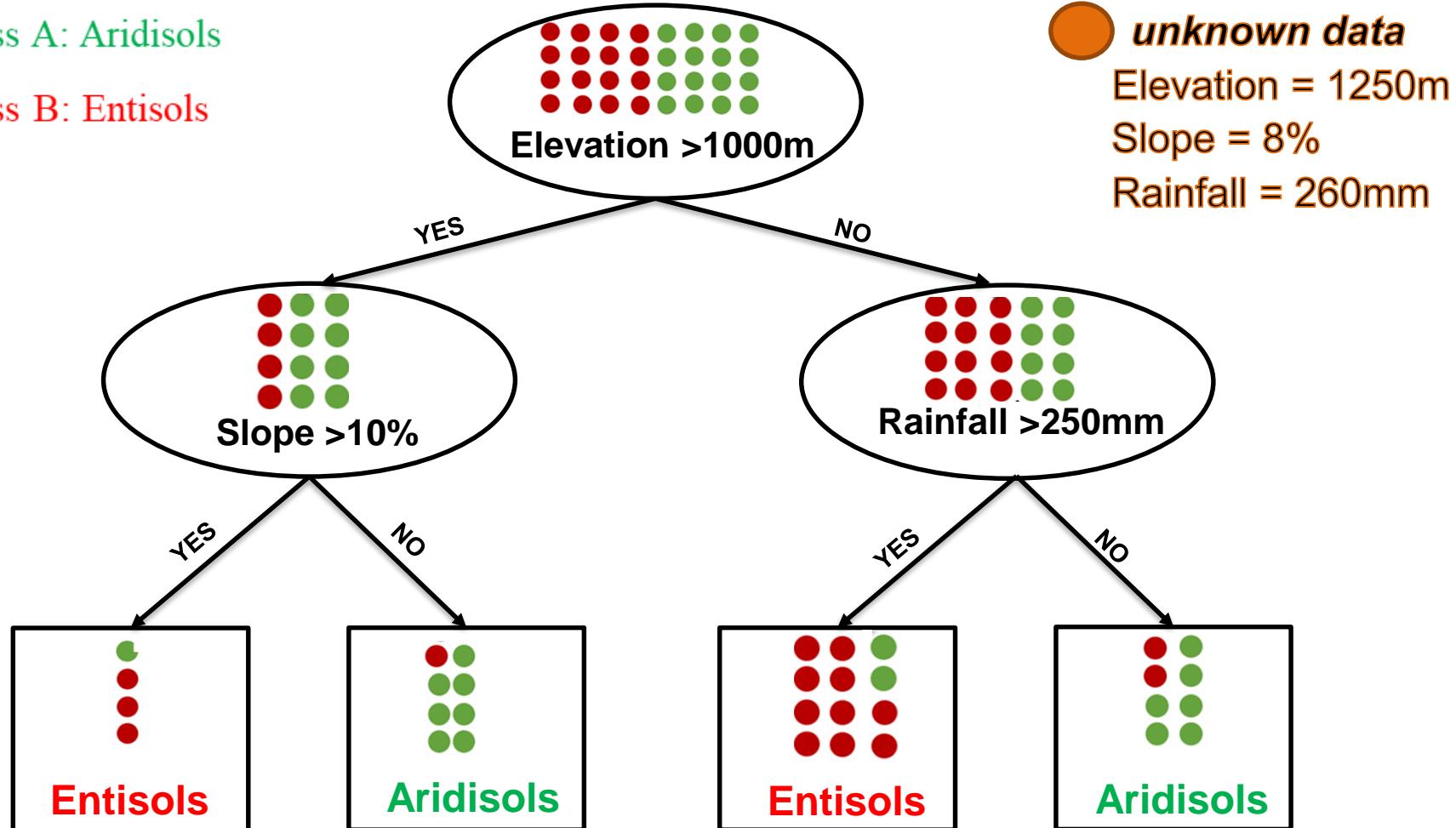
Decision Tree: example II



A decision tree model to predict Entisols and Aridisols

Decision Tree: example II

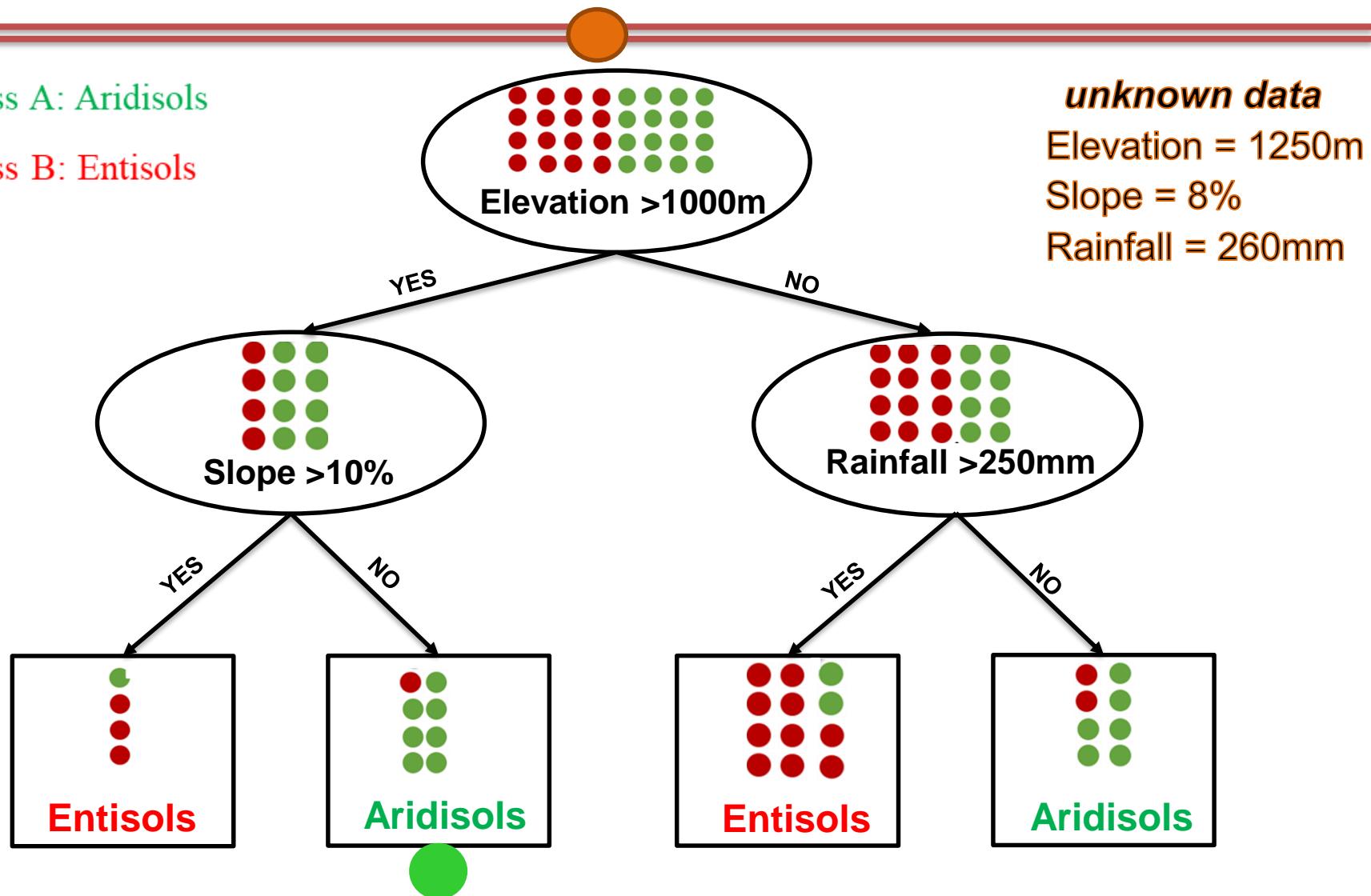
- Class A: Aridisols
- Class B: Entisols



A decision tree model to predict Entisols and Aridisols

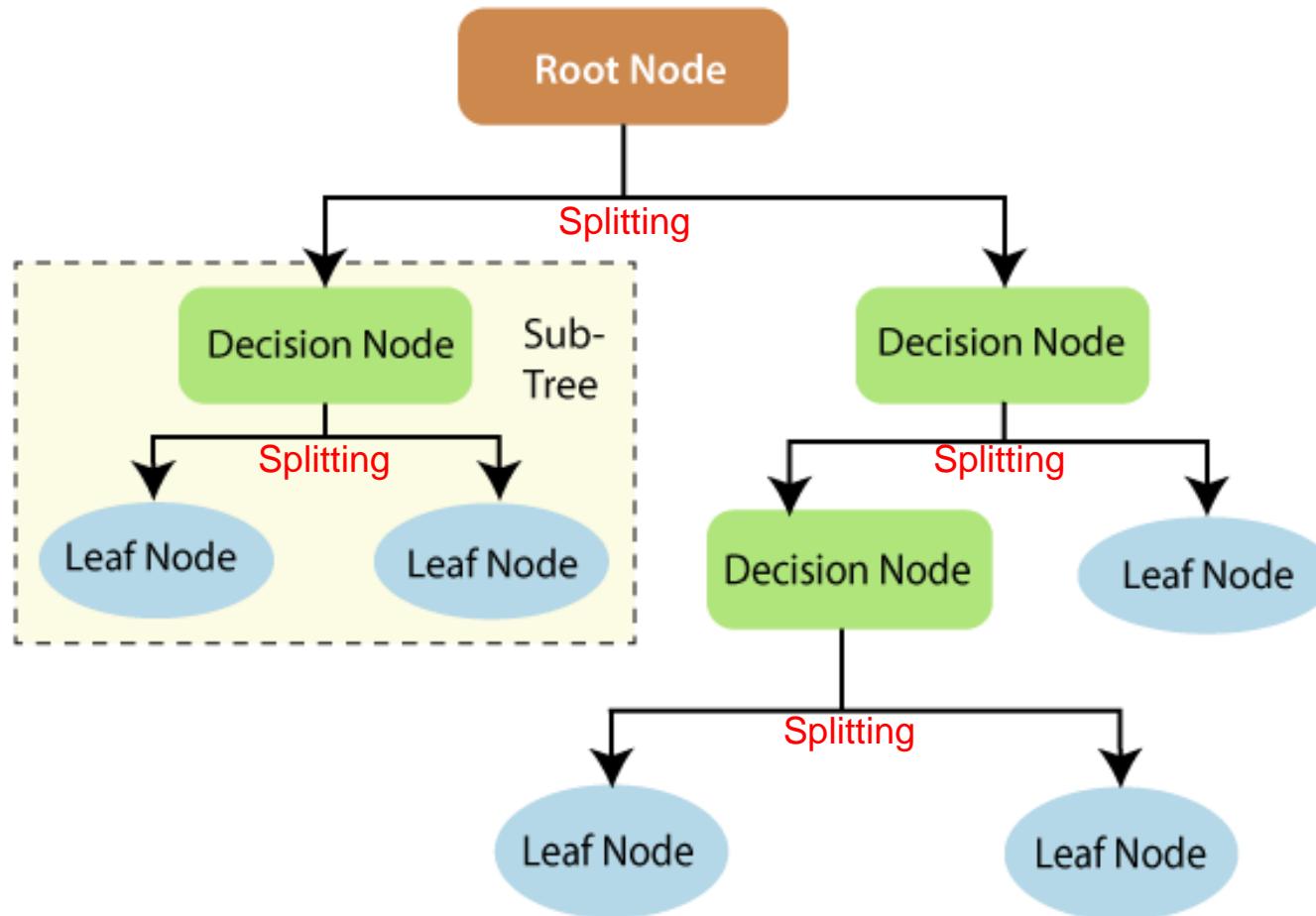
Decision Tree: example II

- Class A: Aridisols
- Class B: Entisols



A decision tree model to predict Entisols and Aridisols

Decision Tree: graphic



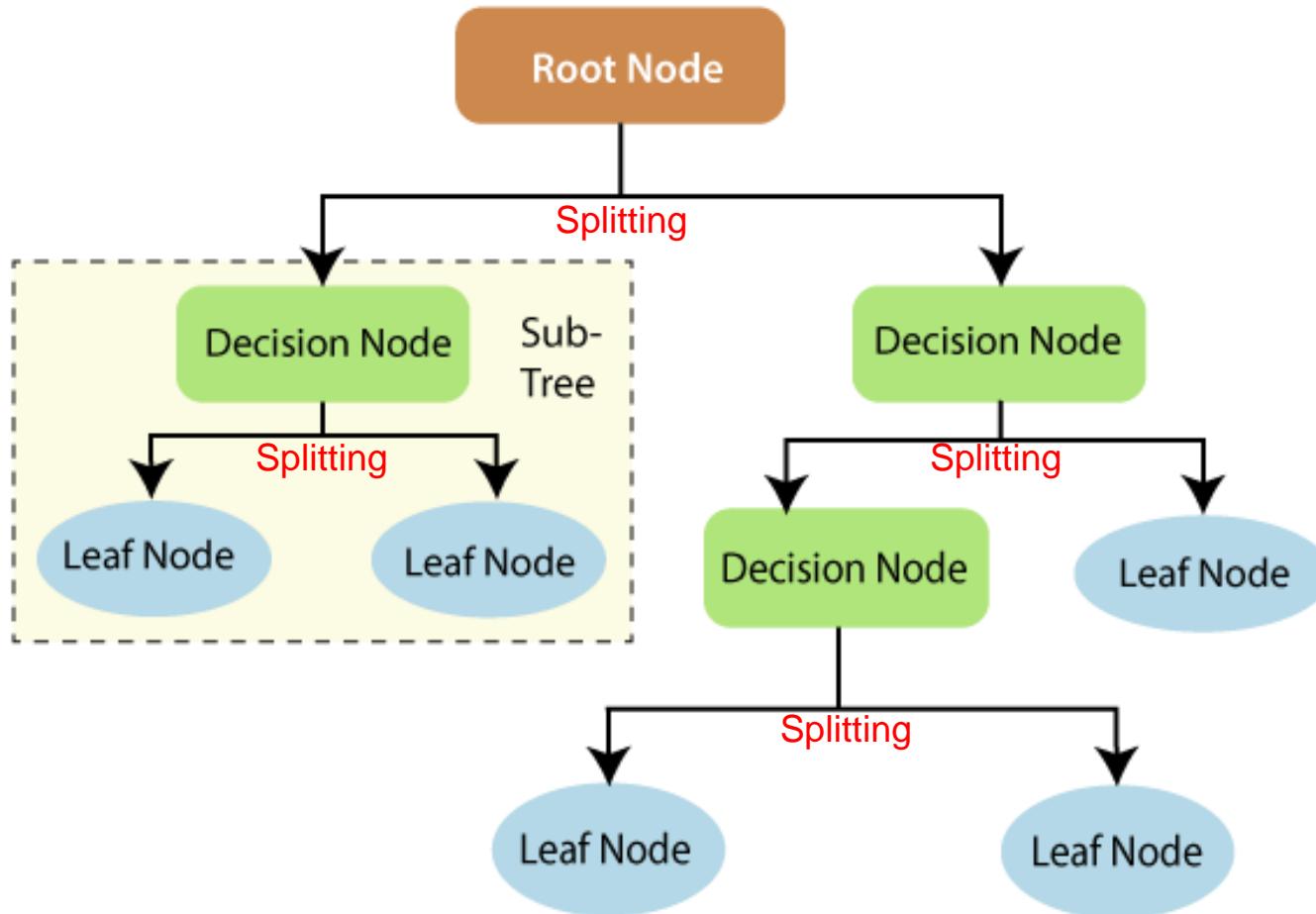
Decision Tree: Terminology

- **Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called the decision node.
- **Leaf/Terminal Node:** Nodes do not split is called Leaf or Terminal node.
- **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.
- **Splitting:** It is a process of dividing a node into two or more sub-nodes.
- **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
- **Branch/Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.

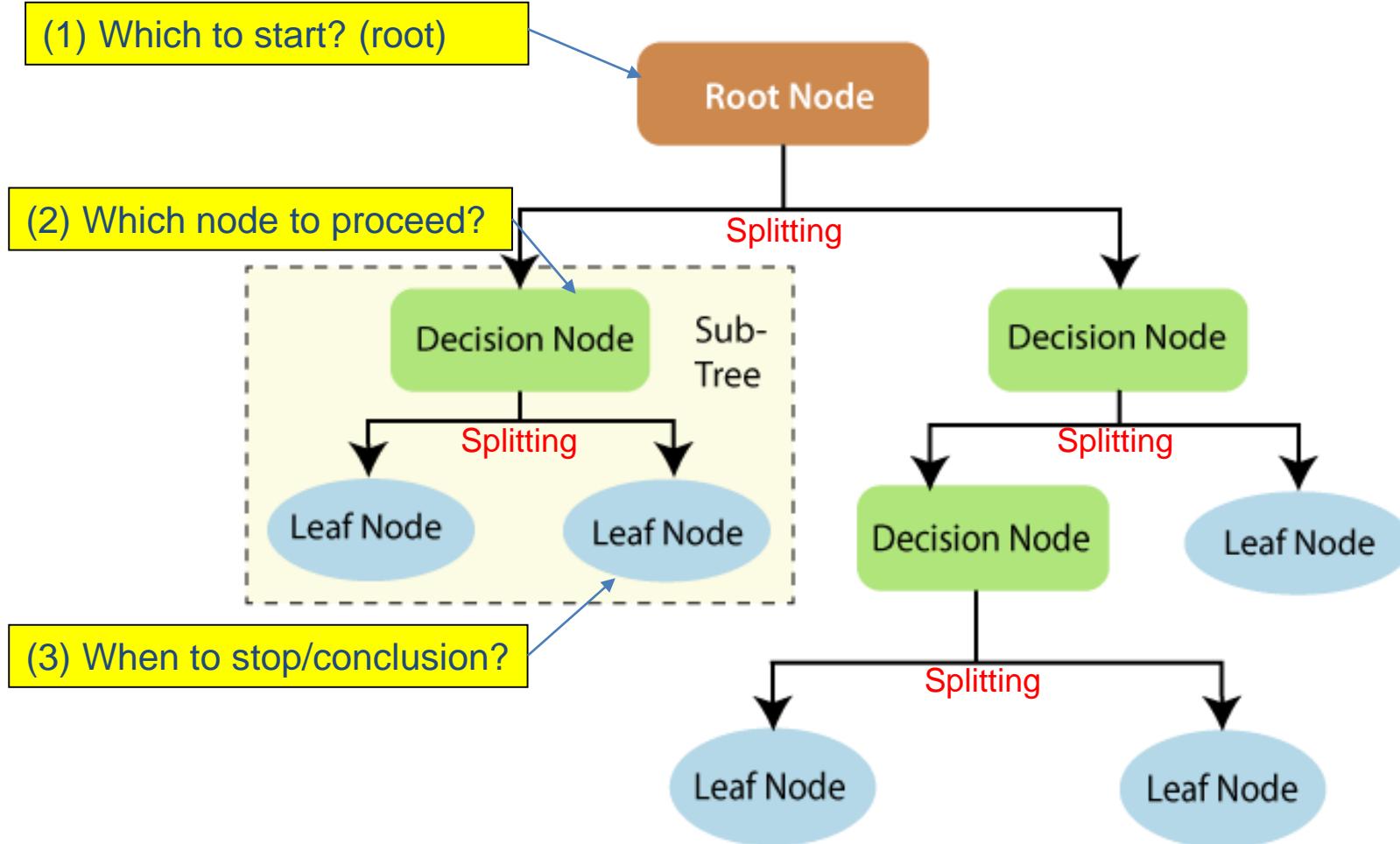
Decision Tree: Terminology

- **Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called the decision node.
- **Leaf/Terminal Node:** Nodes do not split is called Leaf or Terminal node.
- **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.
- **Splitting:** It is a process of dividing a node into two or more sub-nodes.
- **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
- **Branch/Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.

Splitting Issues



Splitting Issues



Splitting Issues

Issues

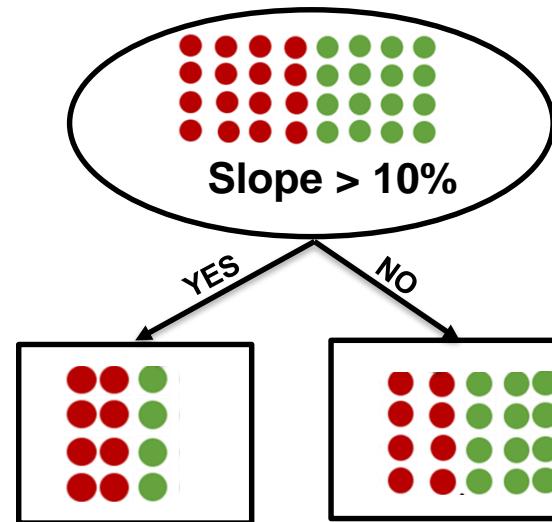
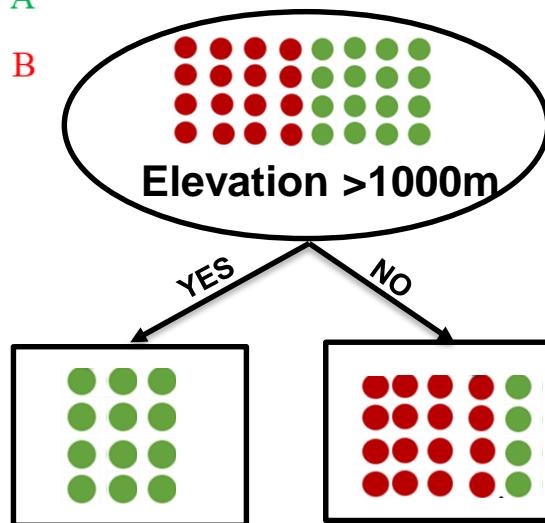
- How to determine the Best Split
- Determine when to stop splitting

How to determine the Best Split

- at each step, we look for **the best split** (Greedy Strategy).



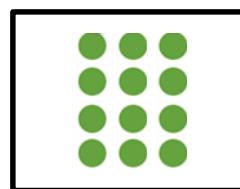
Class A
Class B



Which test split is the best?

How to determine the Best Split

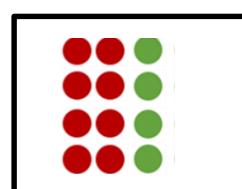
- Nodes with **homogeneous** class distribution are preferred
- Need a measure of node **impurity**:



● Class A
● Class B

12
0

Homogeneous,
Low degree of impurity



● Class A
● Class B

4
8

Non-homogeneous,
High degree of impurity

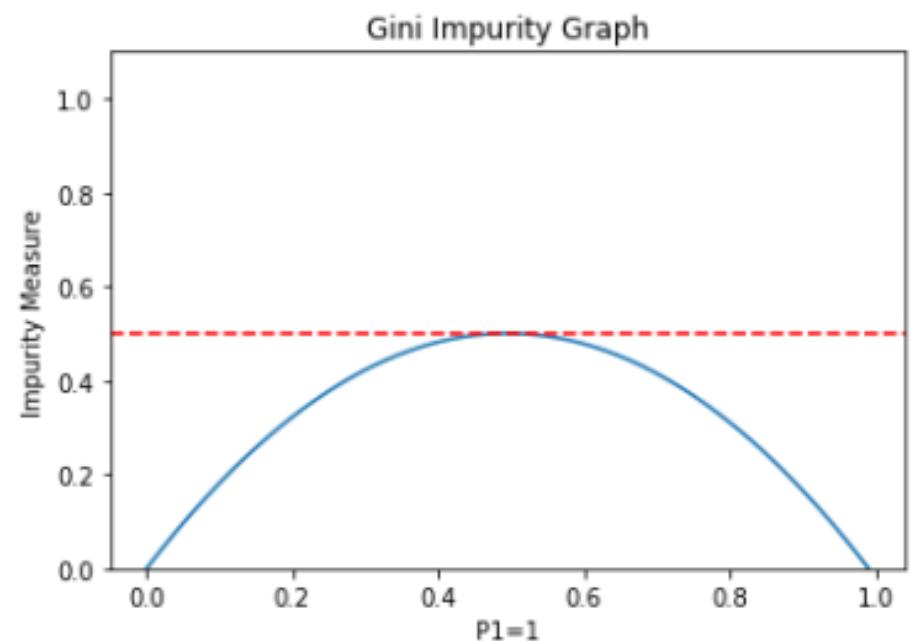
Measures of Node Impurity

- Gini Index or Gini ratio
- This measure provides a ranking to the attributes for splitting the data.
- It stores sum of squared probabilities of each class. We can formulate it as illustrated below.

$$Gini = 1 - \sum (P_i)^2$$

$i = 1$ to number of classes
 P = probabilities of each class

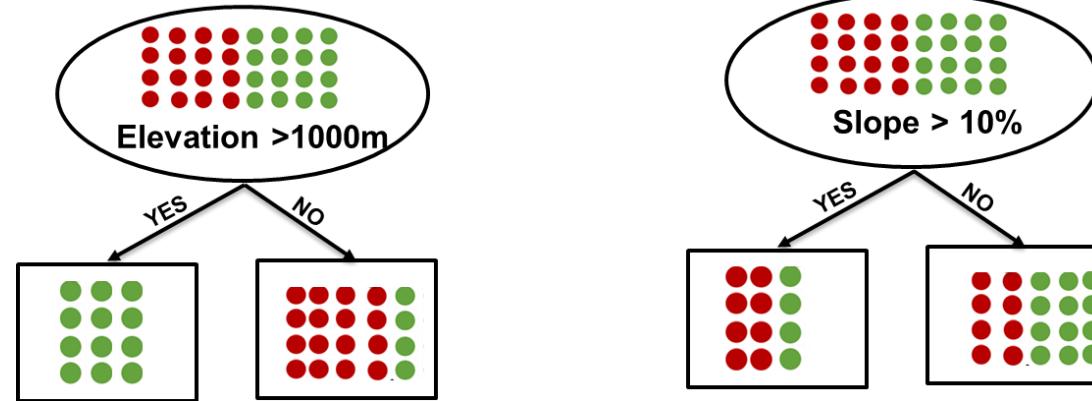
A Gini Impurity of **0** is the lowest.
It can only be achieved when everything is the same class



$$Gini = 1 - \sum(P_i)^2$$

Gini Index

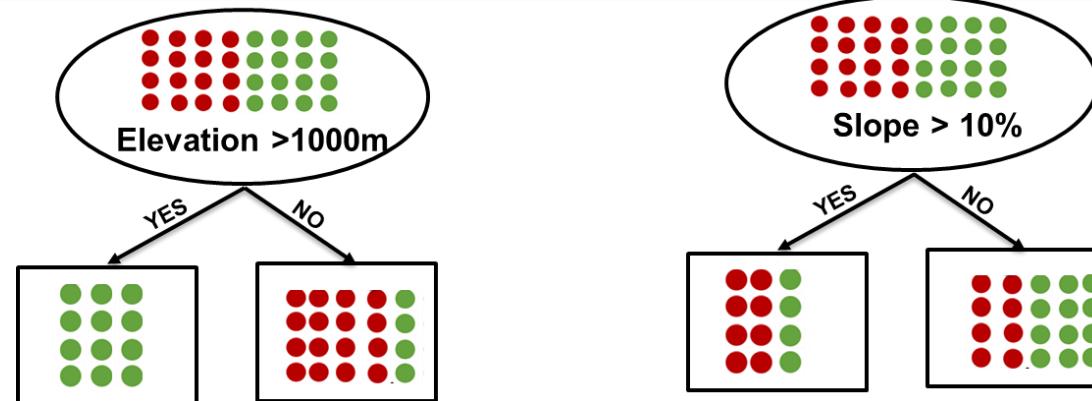
Class A
Class B



$$Gini = 1 - \sum(P_i)^2$$

Gini Index

● Class A
● Class B

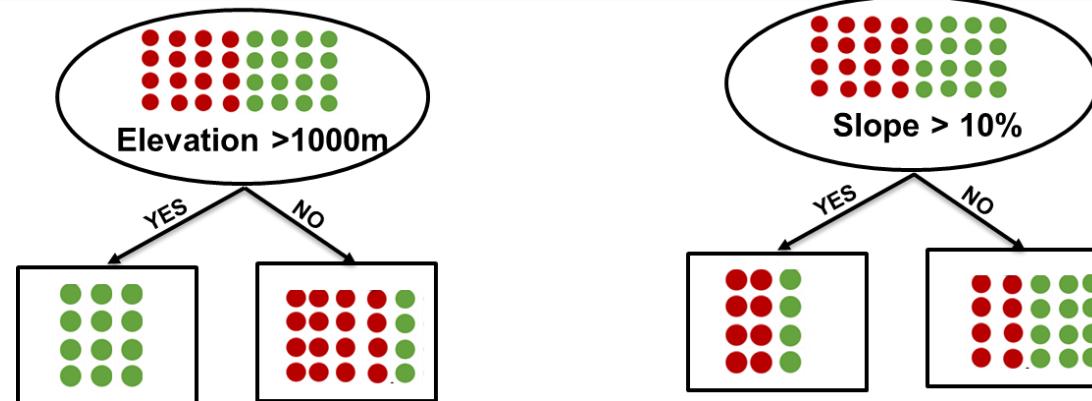


Elevation	A	B	P (A)	P (B)	Gini
>1000 m	12	0	12÷12=1	0÷12=0	1-[(1) ² +(0) ²]=0
<1000 m	4	16	4÷20=0.2	16÷20=0.8	1-[(0.2) ² +(0.8) ²]=0.32

$$Gini = 1 - \sum(P_i)^2$$

Gini Index

Class A
Class B



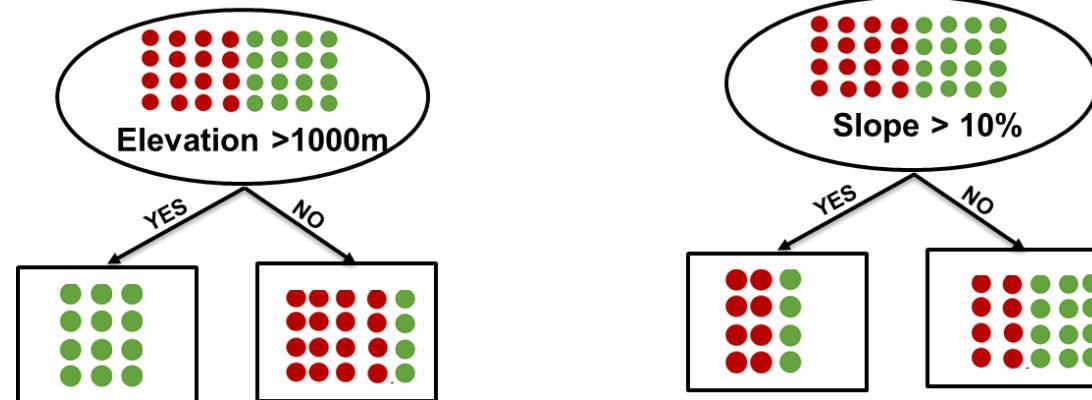
Elevation	A	B	P (A)	P (B)	Gini
>1000 m	12	0	12÷12=1	0÷12=0	1-[(1) ² +(0) ²]=0
<1000 m	4	16	4÷20=0.2	16÷20=0.8	1-[(0.2) ² +(0.8) ²]=0.32

$$Gini(Elevation) = \left(\frac{12}{32}\right) \times 0 + \left(\frac{20}{32}\right) \times 0.32 = 0.20$$

$$Gini = 1 - \sum(P_i)^2$$

Gini Index

Class A
Class B



Elevation	A	B	P (A)	P (B)	Gini
>1000 m	12	0	12÷12=1	0÷12=0	1-[1^2+0^2]=0
<1000 m	4	16	4÷20=0.2	16÷20=0.8	1-[0.2^2+(0.8)^2]=0.32

$$Gini(Elevation) = \left(\frac{12}{32}\right) \times 0 + \left(\frac{20}{32}\right) \times 0.32 = 0.20$$

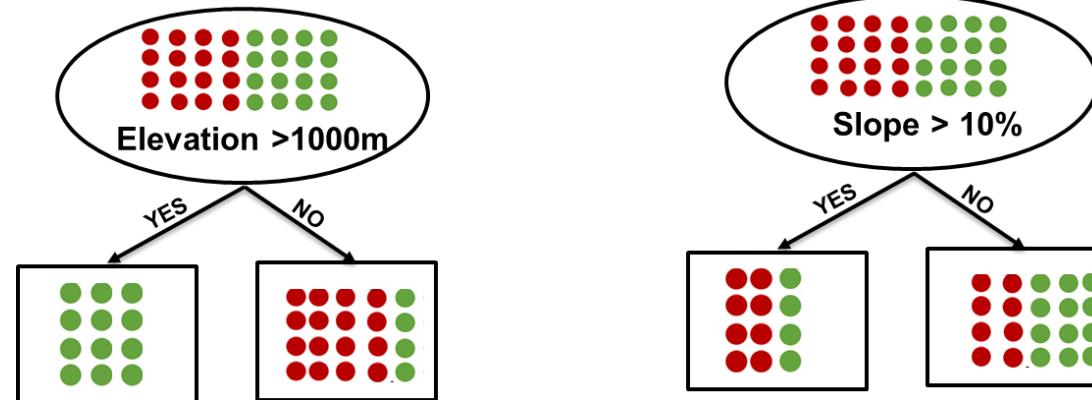
Slope	A	B	P (A)	P (B)	Gini
>10 %	4	8	4÷12=1	8÷12=0	1-[0.22^2+(0.66)^2]=0.44
<10 %	12	8	12÷20=0.2	8÷20=0.8	1-[0.6^2+(0.4)^2]=0.48

$$Gini(Slope) = \left(\frac{12}{32}\right) \times 0.44 + \left(\frac{20}{32}\right) \times 0.48 = 0.46$$

$$Gini = 1 - \sum(P_i)^2$$

Gini Index

Class A
Class B



Elevation	A	B	P (A)	P (B)	Gini
>1000 m	12	0	12÷12=1	0÷12=0	1-[(1) ² +(0) ²]=0
<1000 m	4	16	4÷20=0.2	16÷20=0.8	1-[(0.2) ² +(0.8) ²]=0.32

$$Gini(Elevation) = \left(\frac{12}{32}\right) \times 0 + \left(\frac{20}{32}\right) \times 0.32 = 0.20$$

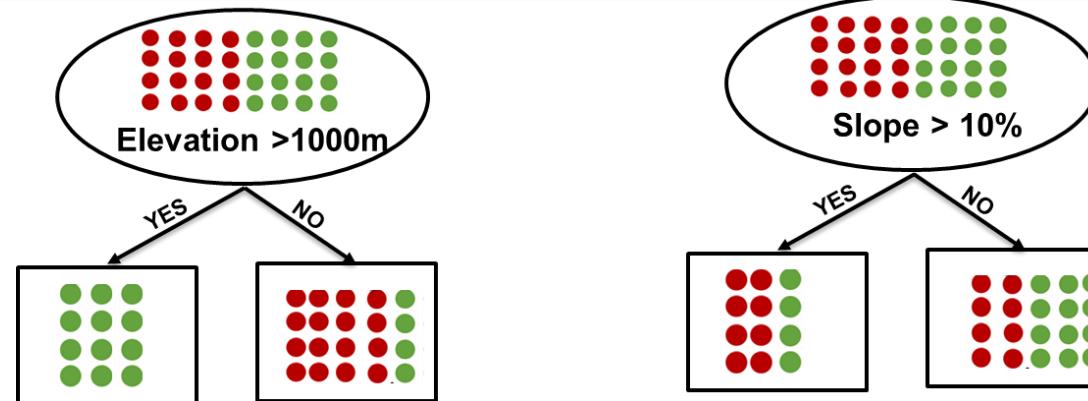
Slope	A	B	P (A)	P (B)	Gini
>10 %	4	8	4÷12=1	8÷12=0	1-[(0.22) ² +(0.66) ²]=0.44
<10 %	12	8	12÷20=0.2	8÷20=0.8	1-[(0.6) ² +(0.4) ²]=0.48

$$Gini(Slope) = \left(\frac{12}{32}\right) \times 0.44 + \left(\frac{20}{32}\right) \times 0.48 = 0.46$$

$$Gini = 1 - \sum(P_i)^2$$

Gini Index

Class A
Class B



Elevation	A	B	P (A)	P (B)	Gini
>1000 m	12	0	12÷12=1	0÷12=0	1-[(1) ² +(0) ²]=0
<1000 m	4	16	4÷20=0.2	16÷20=0.8	1-[(0.2) ² +(0.8) ²]=0.32



$$Gini(Elevation) = \left(\frac{12}{32}\right) \times 0 + \left(\frac{20}{32}\right) \times 0.32 = 0.20$$

Slope	A	B	P (A)	P (B)	Gini
>10 %	4	8	4÷12=1	8÷12=0	1-[(0.22) ² +(0.66) ²]=0.44
<10 %	12	8	12÷20=0.2	8÷20=0.8	1-[(0.6) ² +(0.4) ²]=0.48

$$Gini(Slope) = \left(\frac{12}{32}\right) \times 0.44 + \left(\frac{20}{32}\right) \times 0.48 = 0.46$$

Splitting Issues

Issues

- How to determine the Best Split
- Determine when to stop splitting

Determine when to stop splitting

- **Stopping Rule**
 - Typical stopping conditions for a node:
 - Stop if all instances belong to the same class
 - Stop if all the attribute values are the same
 - More restrictive conditions:
 - Stop if number of instances is less than some user-specified threshold
 - Stop if class distribution of instances are independent of the available features
 - Stop if expanding the current node does not improve impurity measures (e.g., Gini index).

Stop the algorithm before it becomes a fully-grown tree (**Pre-Pruning**)

Overfitting In Decision Trees

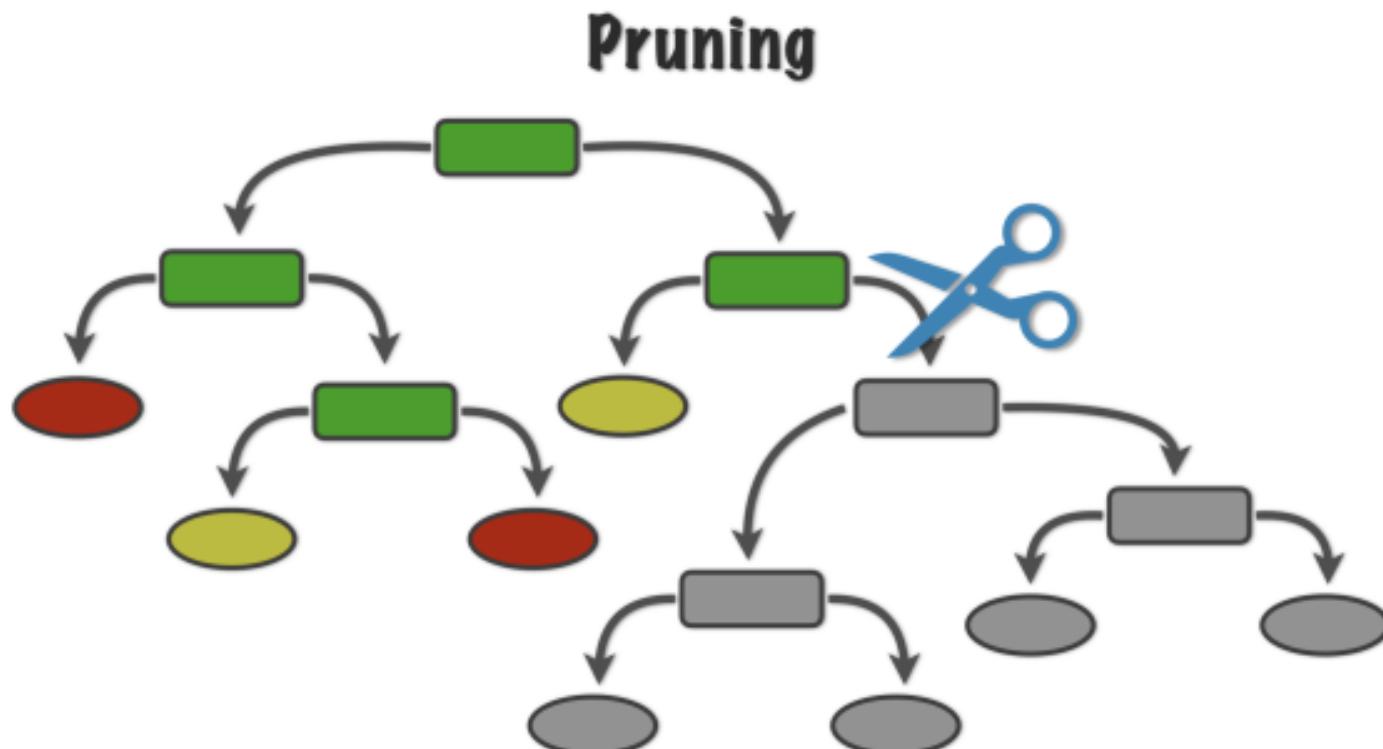
Overfitting happens when a decision tree tries to be as perfect as possible by increasing the depth of tests and thereby reduces the error.

- Post-pruning
 - Grow decision tree to its entirety
 - Trim the nodes of the decision tree in a bottom-up fashion
 - If generalization error improves after trimming, replace sub-tree by a leaf node.
 - Class label of leaf node is determined from majority class of instances in the sub-tree

Pruning

Comparing pre-pruning and post-pruning,

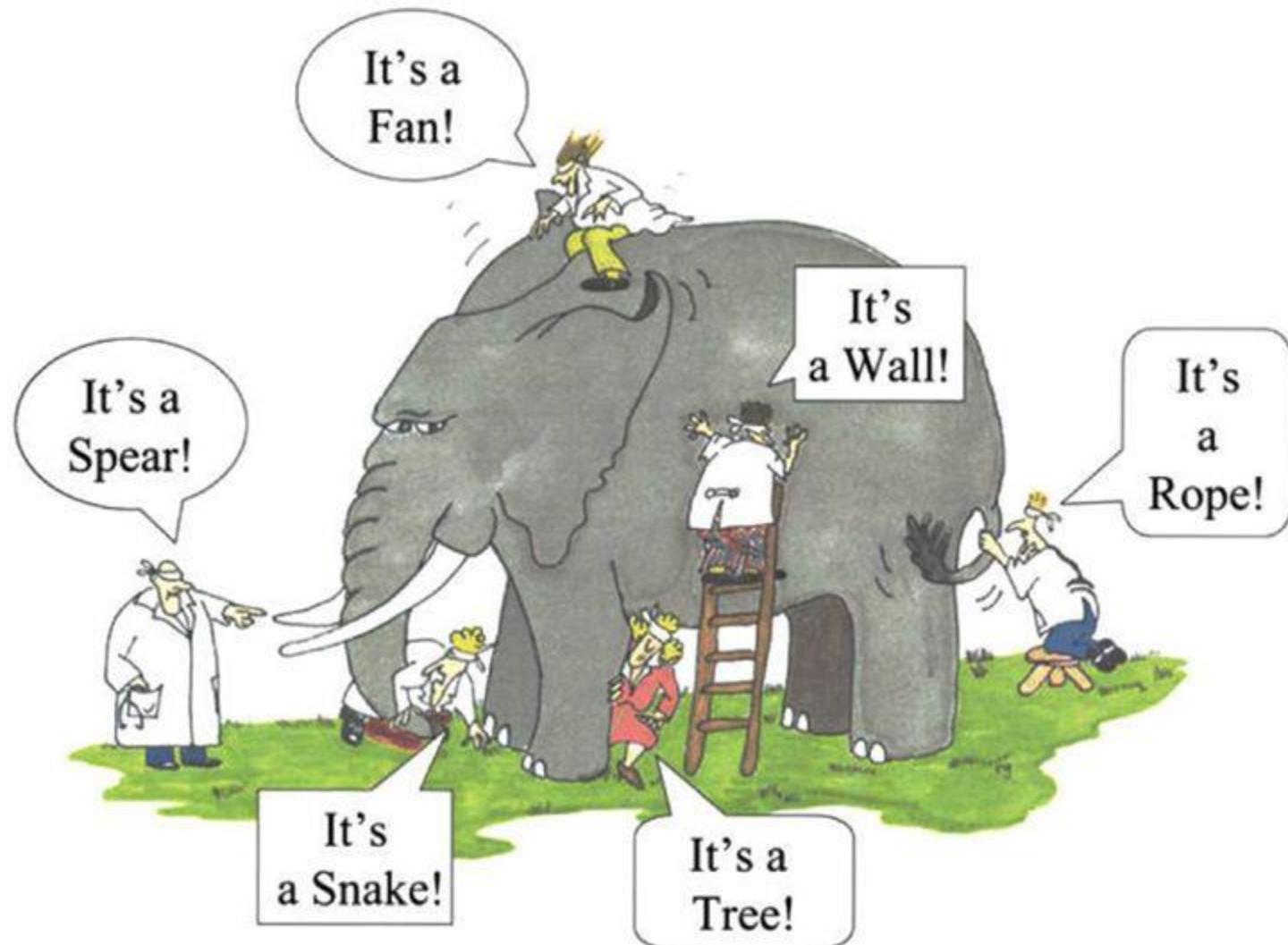
- Pre-pruning is faster but
- Post-pruning generally leads to more accurate trees.



Advantages and Disadvantages

- **Advantages:**
 - Extremely fast at classifying unknown records.
 - Simple to understand and to interpret for small-sized trees
 - Trees can be visualized.
 - Excludes unimportant features.
- **Disadvantages:**
 - Easy to overfit.
 - Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an **ensemble**.

Ensemble Methods

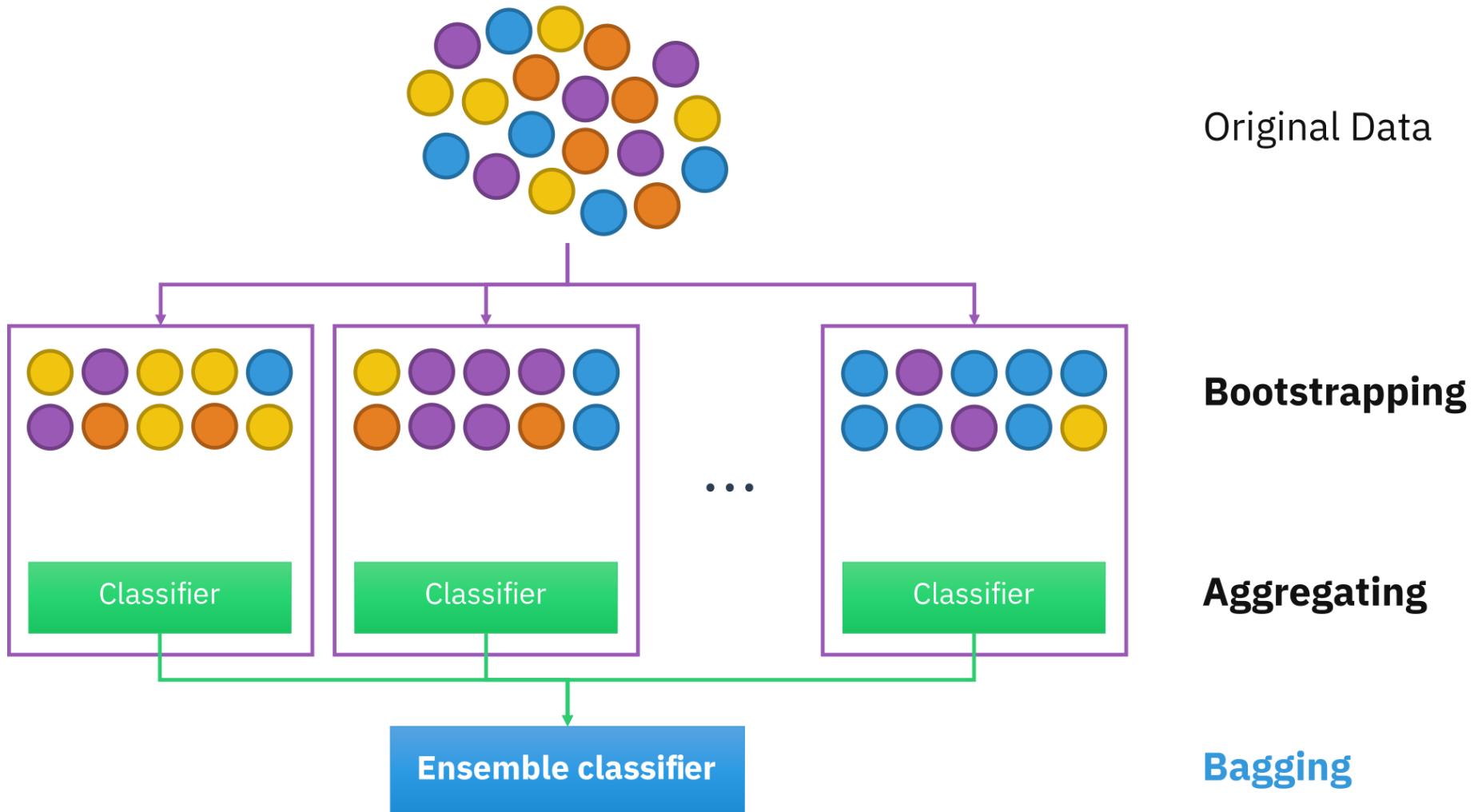


Bagging

- Bagging or Bootstrap aggregating (Breiman 1994):
- First, we **create multiple bootstrap samples** so that each new bootstrap sample will act as another (almost) independent dataset drawn from true distribution.
- Then, we can fit a weak learner (**decision tree**) for each of these samples
- Finally **aggregate them** (simple average for regression or simple majority vote for classification)

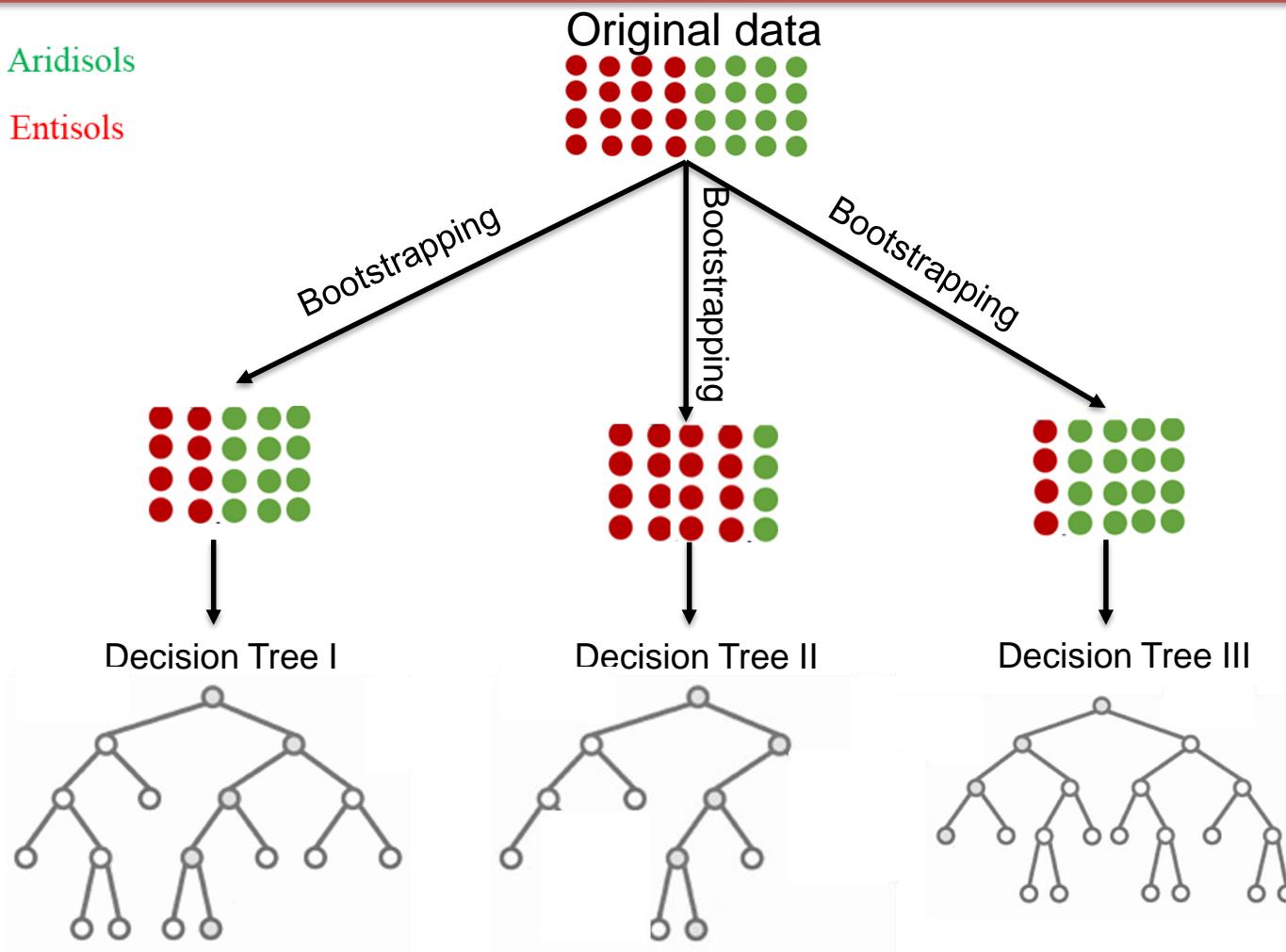
Bootstrap: a statistical technique, for generating samples of **size B** (called bootstrap samples) from an initial dataset of **size N** by randomly drawing with replacement B observations.

Bagging



Bagging: example I

- Class A: Aridisols
- Class B: Entisols



Bagging: example I

- Class A: Aridisols
- Class B: Entisols

Original data



unknown data
Elevation = 1250m
Slope = 8%
Rainfall = 260mm

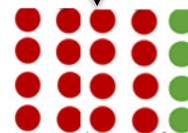
Bootstrapping

Bootstrapping

Bootstrapping



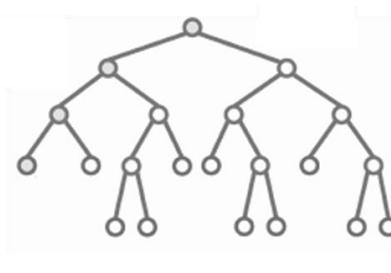
Decision Tree I



Decision Tree II

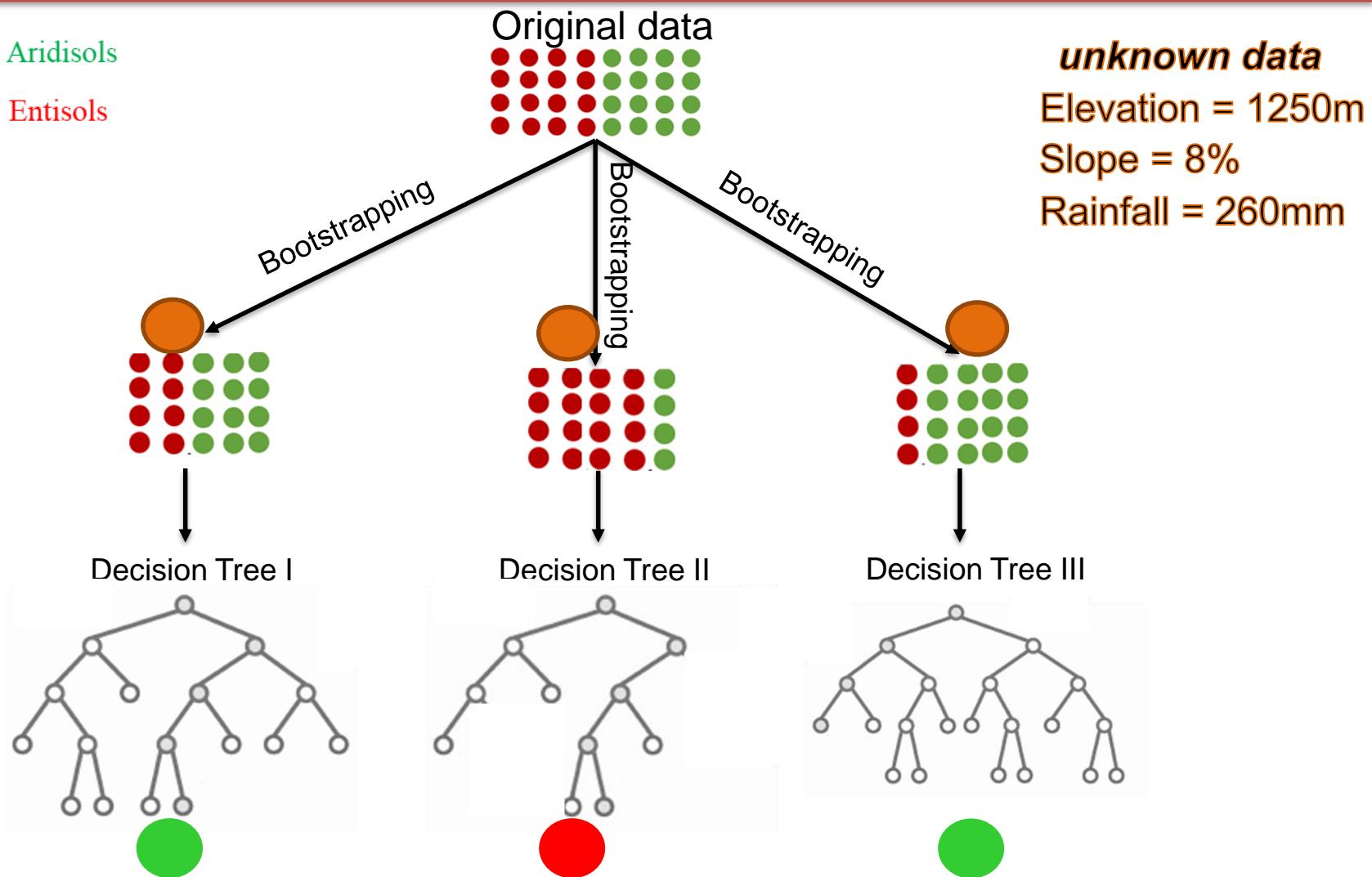


Decision Tree III

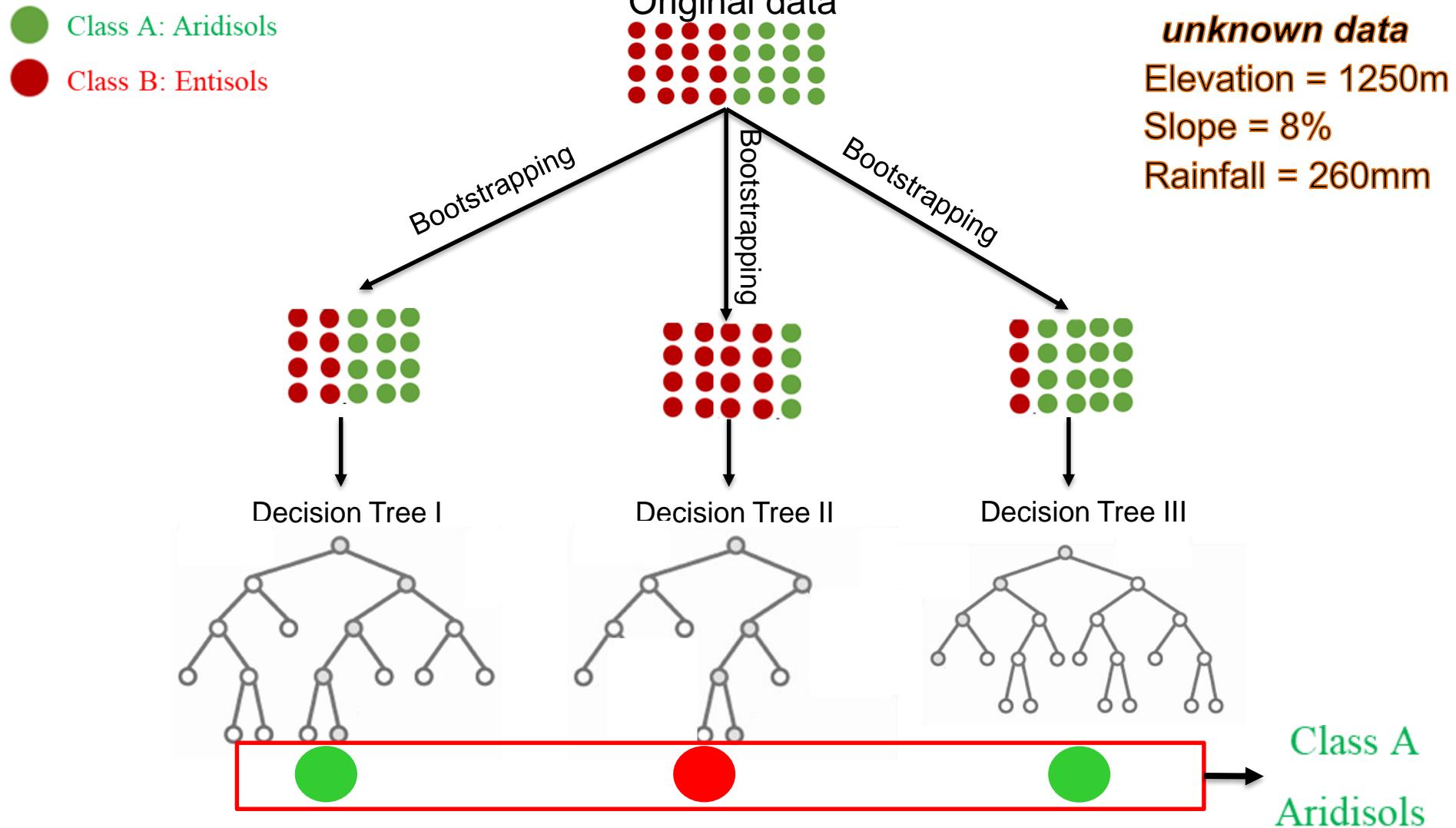


Bagging: example I

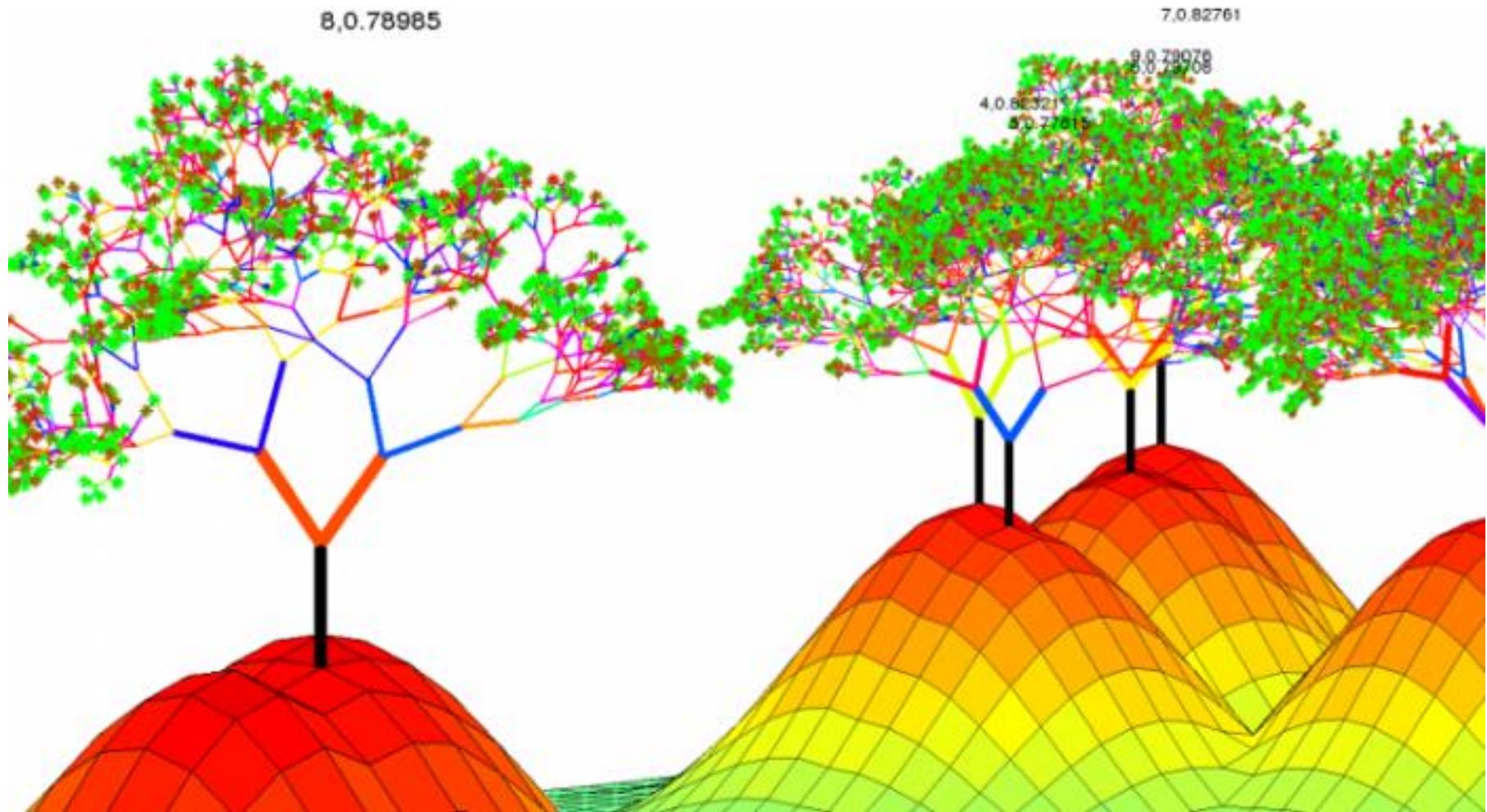
- Class A: Aridisols
- Class B: Entisols



Bagging: example I



Random Forest



Random Forest

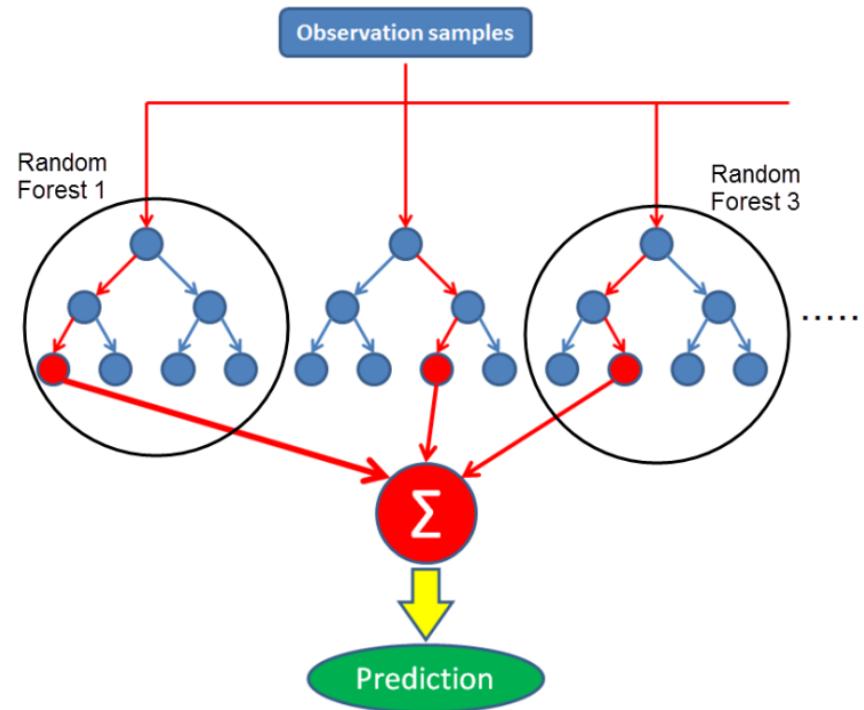
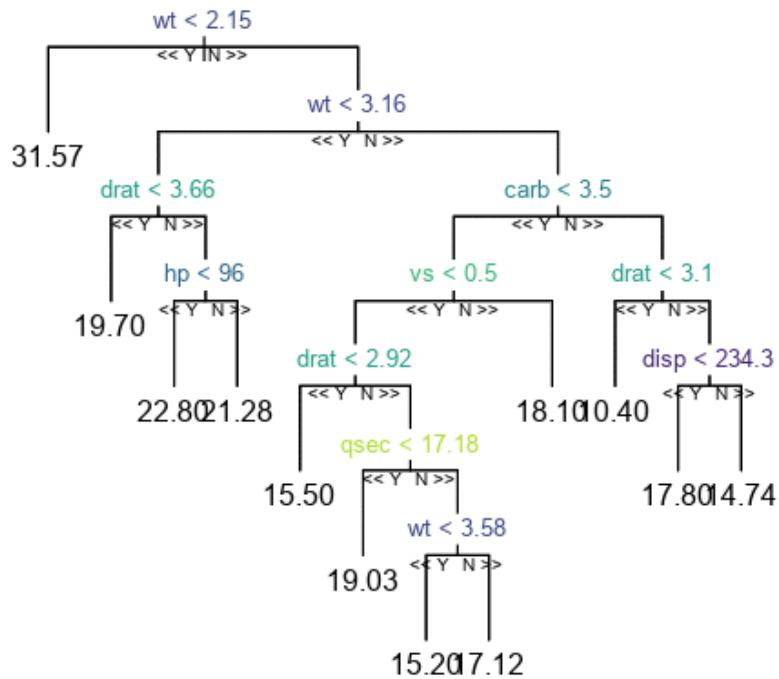
- **Random Forest:**
 - The random forest approach is a **bagging method** for classification and regression that operate by constructing multiple decision trees at training phase and outputting the class that is the mode of the classes or mean/average prediction of the individual trees.
 - Random forests also use another trick to make the multiple decision trees a bit less correlated with each others:
 - when growing each tree, instead of **only** sampling over the observations in the dataset to generate a bootstrap sample, we also sample over **covariates** and keep only a random subset of them to build the tree.

How Does It Work?

- **Random Forest:**
 - Draw a bootstrap sample:
 - Random selection of 2/3 of the training data; repeat n times.
 - Grow an unpruned tree to each bootstrap sample
 - Random predictor selection: for each split in each tree a random subset is selected from the predictor variables. The best split is chosen from among the selected predictors.
 - Predict new data by aggregating the predictions of the n trees.
 - Average for continuous variables
 - Majority vote for categorical variables

Random Forest

Tree 1



Out Of Bag Accuracy Assessment

- **Random Forest:**
 - Random Forest comes with an internal accuracy assessment (based on cross validation).
 - Bootstrap sampling: the algorithm sets aside ~36% of the training data for each tree grown: out of bag data (OOB).
 - OOB data can be used to asses prediction accuracy:
 - Predict the data not in the bootstrap sample for each tree
 - Aggregate the OOB predictions: mean (continuous data), majority vote (categorical data).

Variable Importance

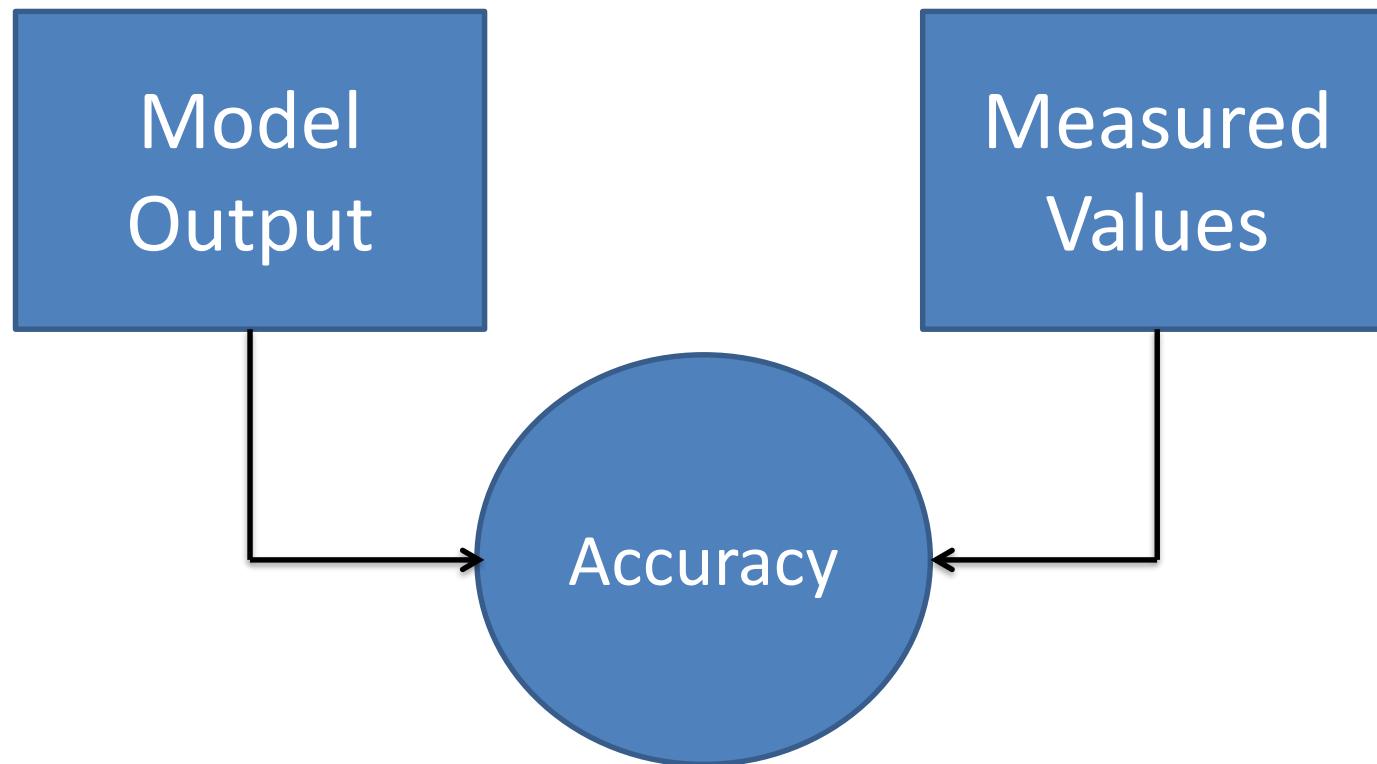
- Random Forest:
 - Ensembles of trees are not easy to interpret.
 - Ensemble can reflect the potentially (complex) effect of a variable on the response
 - Variable importance plot: shows how much prediction error increases when the values of one predictor are permuted (break association with response) while all others are left unchanged.
 - Permuted variable is used together with other variables to predict the response, **prediction accuracy will decrease**.

Advantages and Disadvantages

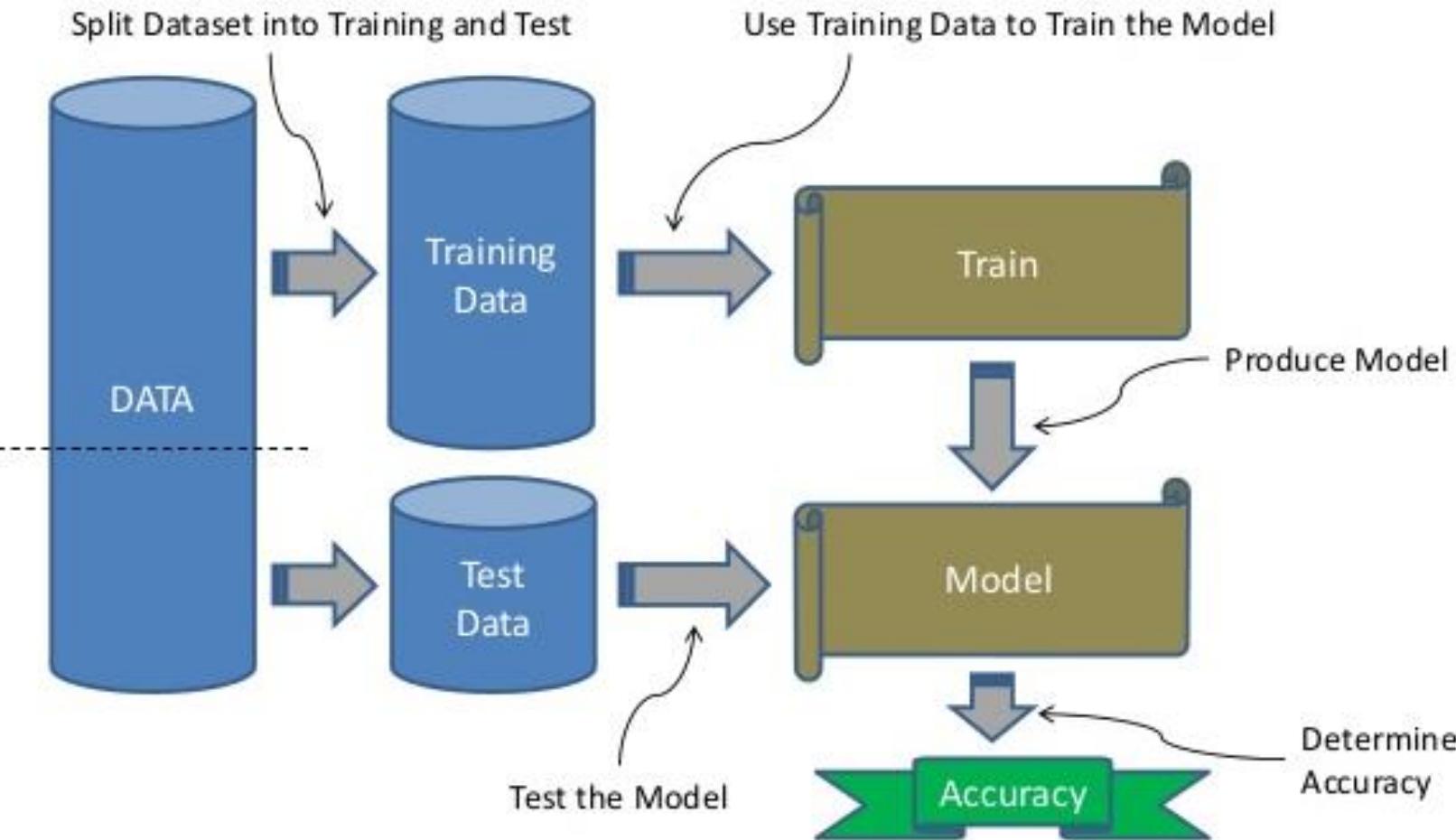
- **Advantages:**
 - Quite fast
 - Models interactions in the data / non linear relationships
 - Very good predictive power expected
 - Yields covariate importance, makes covariate selection possible
- **Disadvantages:**
 - Difficult to interpret

Accuracy Assessment of Models

- Assess accuracy of a model output is one of the most important steps in digital soil mapping



Training and Testing the Models



Performance Metrics for Regression

- Mean Absolute Error:

$$MAE = \frac{1}{n} \sum_{\text{Sum of}} |y - \hat{y}|$$

Divide by the total number of data points

Predicted output value

Actual output value

The absolute value of the residual

- Root Mean Square Error:

$$RMSE = \sqrt{\sum_{i=1}^n \frac{(\hat{y}_i - y_i)^2}{n}}$$

Practice

Import covariates

Import point data

Overlay point data on covariates

Make a geodatabase

Split the geodatabase to training and testing sets

Train machine learning models

Test machine learning models

Predict soil maps



1. Open Rstudio

The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with various icons. A script editor window titled "Untitled1*" contains the following R code:

```
1 1 + 1
2 2 + 3 + 4
3 x <- c(1:100)
4 hist(x)
5
```

The console window displays the standard R startup message, which includes:

You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

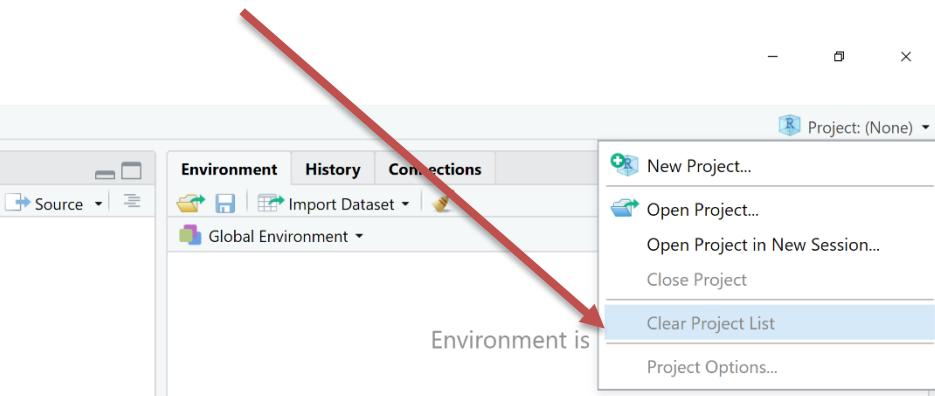
Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

A search bar at the bottom says "Type here to search".

2. Clear Project List



3. New Project

The screenshot shows the RStudio interface. In the top right corner, a red arrow points to the 'Project' dropdown menu, which is open to show options like 'New Project...', 'Open Project...', and 'Clear Project List'. The 'Environment' tab is selected in the main workspace. On the left, there's an 'Untitled1*' script editor with some R code. Below it is the 'Console' window displaying the standard R welcome message. The bottom status bar shows the date and time.

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Untitled1* Go to file/function Addins

1 1 + 1
2 2 + 3 + 4
3 x <- c(1:100)
4 hist(x)
5

5:1 (Top Level) R Script

Console ~/
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Type here to search

Project: (None)

New Project...
Open Project...
Open Project in New Session...
Close Project
Clear Project List
Project Options...

Environment History Connections

Import Dataset

Global Environment

Files Plots Packages Help Viewer

4:35 PM 2/9/2021

4. Existing Directory

The screenshot shows the RStudio interface with a 'Create Project' dialog box overlaid. The dialog has three options: 'New Directory', 'Existing Directory', and 'Version Control'. A large red arrow points to the 'Existing Directory' option. The RStudio environment includes a top menu bar, a code editor with some R code, a console window displaying a welcome message, and an empty 'Environment' tab.

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Untitled1* Go to file/function Addins

Project: (None)

1 1 + 1
2 2 + 3 + 4
3 x <- c(1:100)
4 hist(x)
5

5:1 (Top Level) ◆

Console ~/ ◆
You are welcome to redistribute
Type 'license()' or 'licence()'
Natural language support but r
R is a collaborative project wit
Type 'contributors()' for more i
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |

New Project

Create Project

New Directory Start a project in a brand new working directory >

Existing Directory Associate a project with an existing working directory > ↓

Version Control Checkout a project from a version control repository >

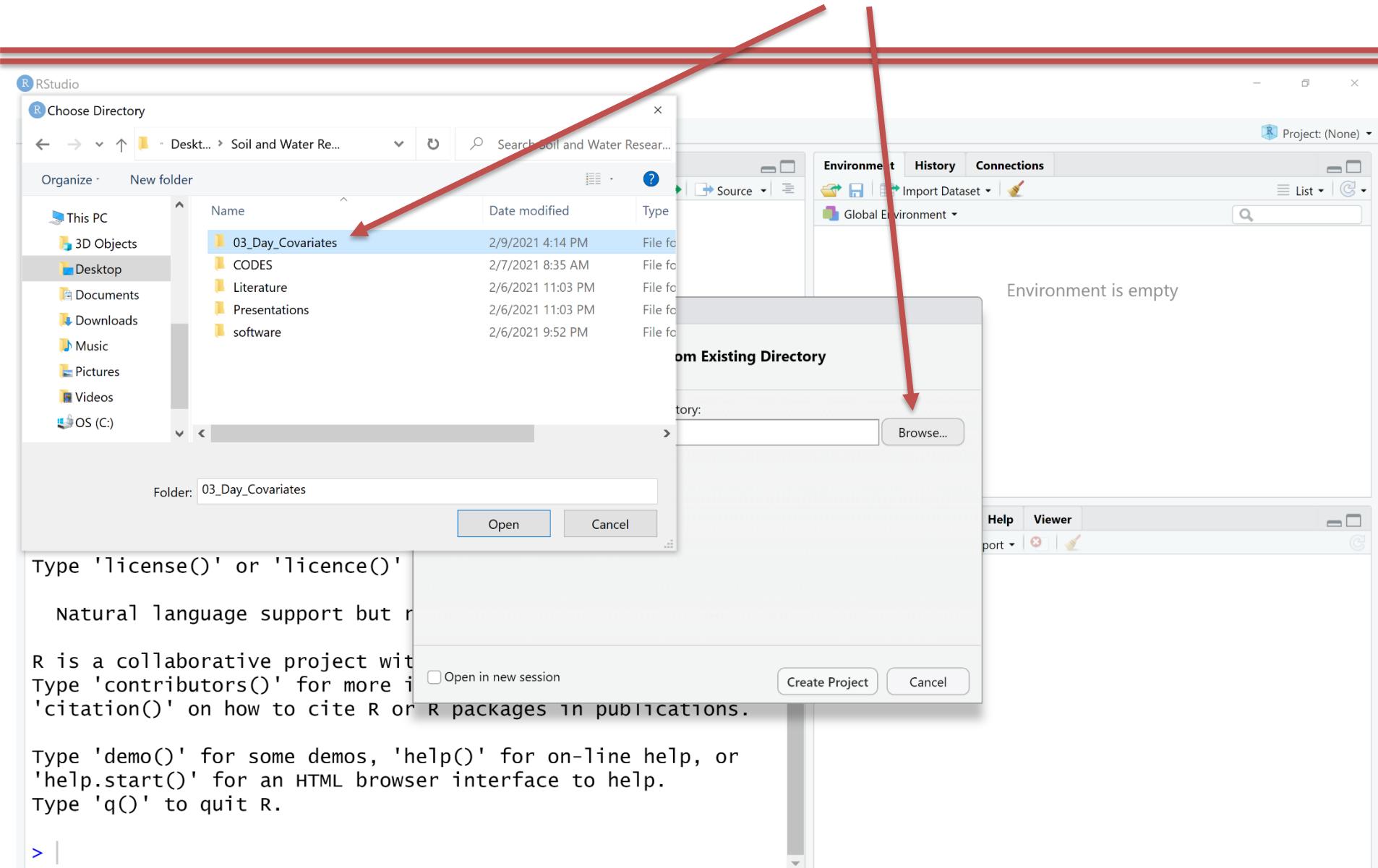
Environment History Connections

Global Environment

Environment is empty

Help Viewer

5. Find the folder and find Day_04_Machine Learning-P1 and open



6. Create project

The screenshot shows the RStudio interface with a red arrow pointing to the 'Create Project' button in the 'Create Project from Existing Directory' dialog box.

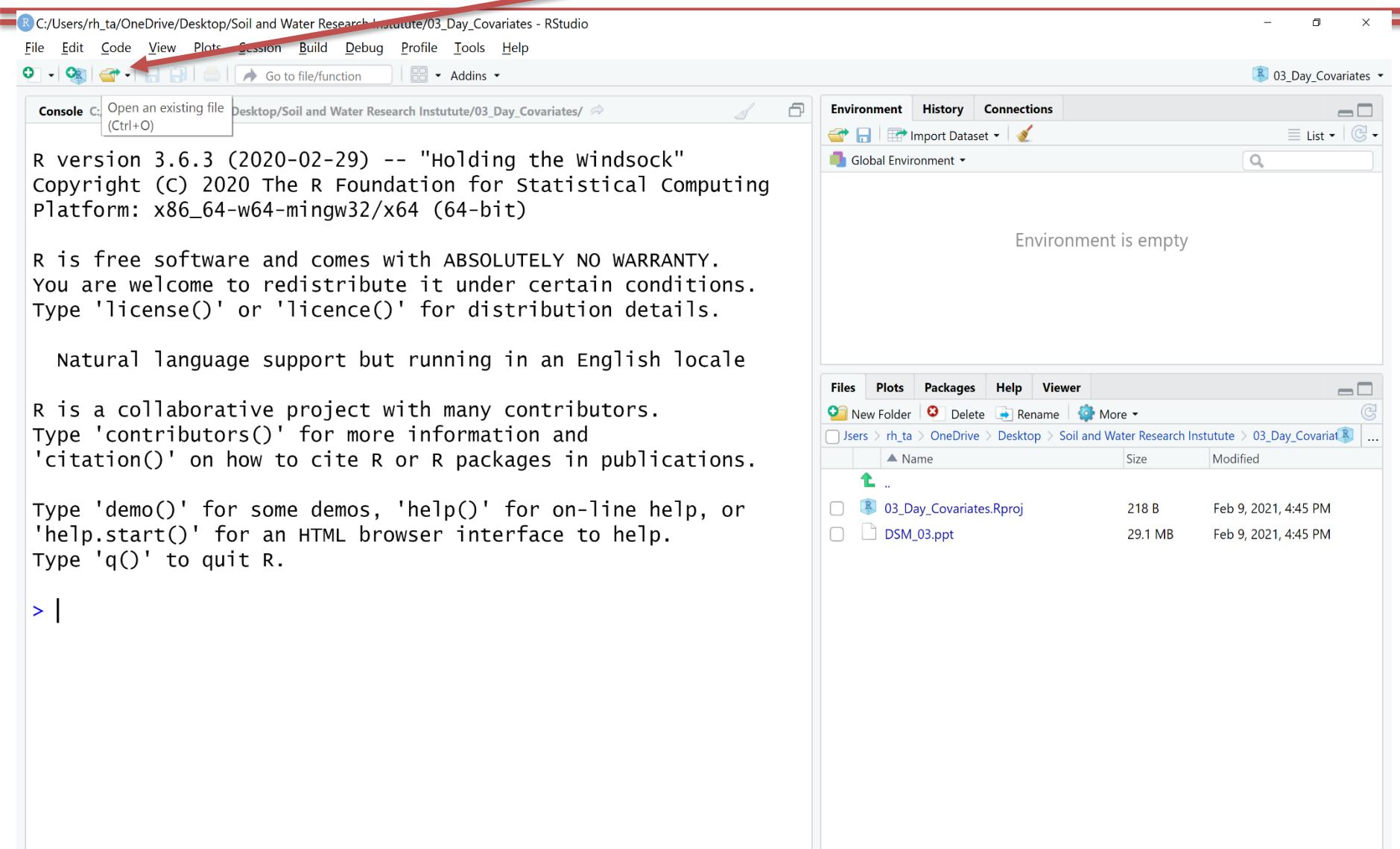
RStudio interface elements:

- File Edit Code View Plots Session Build Debug Profile Tools Help
- Addins
- Untitled1*
- Source on Save
- Run
- Environment History Connections
- Global Environment
- Environment is empty
- Console
- You are welcome to redistribute Type 'license()' or 'licence()'
- Natural language support but r
- R is a collaborative project with Type 'contributors()' for more information 'citation()' on how to cite R or R packages in publications.
- Type 'demo()' for some demos, 'help()' for on-line help, or 'help.start()' for an HTML browser interface to help.
- Type 'q()' to quit R.
- >

New Project dialog box details:

- Create Project from Existing Directory
- Project working directory: C:/Users/rh_ta/OneDrive/Desktop/Oil and Water R
- Back
- Browse...
- Open in new session
- Create Project
- Cancel

7. Open an existing file and find **04_R_Cov.R**



R C:/Users/rh_ta/OneDrive/Desktop/Soil and Water Research Institute/03_Day_Covariates - RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Open an existing file Desktop/Soil and Water Research Institute/03_Day_Covariates/

R version 3.6.3 (2020-02-29) -- "Holding the Windsock"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |

Environment History Connections

Import Dataset

Global Environment

Environment is empty

Files Plots Packages Help Viewer

New Folder Delete Rename More

Name	Size	Modified
03_Day_Covariates.Rproj	218 B	Feb 9, 2021, 4:45 PM
DSM_03.ppt	29.1 MB	Feb 9, 2021, 4:45 PM

8. Open 04_R_Cov.R

