Classification and Regression via Decision Trees and Random Forests - Summary

1. Introduction

This document covers key concepts and techniques for classification and regression using decision trees (DT) and random forests (RF), emphasizing their applications, methodologies, advantages, and disadvantages.

2. Recap/Introduction

Key Points:

- **Supervised vs. Unsupervised Learning:** Supervised learning includes tasks such as classification and regression.
- **Methods Available:** Various statistical models and data-based algorithms like decision trees are used for these tasks.

3. Decision Trees (DT)

What is a Decision Tree?

A decision tree recursively partitions data for classification or regression tasks by making decisions based on whether certain conditions are true or false. Key historical developments include:

- **AID (Automatic Interactive Decision Tree) by Morgan and Sonquist (1963).**
- **High risk of overfitting and lack of analytical rigor initially identified**.

Terminology:

- **Root Node, Node, Terminal Node (Leaves), Branch, Split, Attributes (Features), and Response (Target Variables)**.

Example:

A decision tree can be used to predict whether a person likes Taylor Swift based on age, gender, and income. The process involves creating a root node, determining split conditions, and labeling terminal nodes.

Gini Impurity:

Used to measure the impurity of nodes. For example: Gini=1−(pyes2+pno2)\text{Gini} = 1 - (p_{\text{yes}}^2 + p_{\text{no}}^2)Gini=1−(pyes2+pno2) Calculations are performed to determine the impurity for different splits and nodes.

Splitting Numeric Variables:

Methods include sorting, calculating averages, and using the Gini coefficient to determine optimal split points.

4. Classification and Regression Trees (CART)

Overview:

- **Invented by Leo Breiman and colleagues in 1984.**
- **Handles both continuous and categorical data**.
- **Known for analytical rigor**.

CART Algorithm:

1. Create the root node.
2. Split into child nodes recursively until no further splits are possible.
3. Prune the tree using cost-complexity methods to avoid overfitting.

Key Elements:

1. Selecting splits at intermediate nodes.
2. Determining terminal nodes.
3. Assigning values to terminal nodes.

5. Random Forests (RF)

Overview:

Developed by Leo Breiman around 2000, random forests improve regression results and classification accuracy by using ensembles of trees grown randomly.

Advantages:

- High accuracy, no overfitting, provides variable importance, easily parallelizable, and minimal pre-processing required.

Algorithm:

1. Bootstrap sampling to create training subsets.
2. Random selection of variables at each node.
3. Fully grown trees without pruning.
4. Aggregation of outputs via voting or averaging.

6. Applications and Software

Spatial and Temporal Data:

DT and RF do not directly handle spatio-temporal information but use attribute values at sampled locations and times. It is crucial to validate classifications or regressions spatially.

Software:

- **R Packages:** Party, Rpart, Randomforest.
- **Python Libraries:** Scikit-learn (sklearn).

Conclusion

Decision trees and random forests are powerful tools for classification and regression tasks. They offer significant advantages, such as robustness and ease of interpretation for DTs, and high accuracy and no overfitting for RFs. Despite some disadvantages, their broad applicability and the availability of efficient software tools make them popular choices in machine learning and data analysis.

UNIVERSITY OF TWENTE.

Classification and regression via decision trees and random forests

Raúl ZURITA-MILLA, Shaheen ABDULKAREEN, Rosa AGUILAR

07-Jun-2024

ITC FACULTY OF GEO-INFORMATION SCIENCE AND EARTH OBSERVATION

Contents

- Recap/Introduction
- Decision Trees
  - CART: classification and regression trees
  - Ensembles: Random Forest
- Software
- TBL
- Q/A

Re-cap

- Supervised vs unsupervised learning
- Typical tasks of supervised learning
  - Classification (e.g., land cover maps)
  - Regression/prediction (e.g., biomass maps)

- For theses tasks → many methods available in literature
  - Statistical modeling (e.g., Kriging) vs data-based algorithm (DT)

What is a decision tree?

Decision trees

Decision trees do recursive partitioning of the data for classification and/or regression tasks

- A decision tree makes a statement, and makes a decision based on whether or not that statement is ***True*** or ***False***.

A bit of history

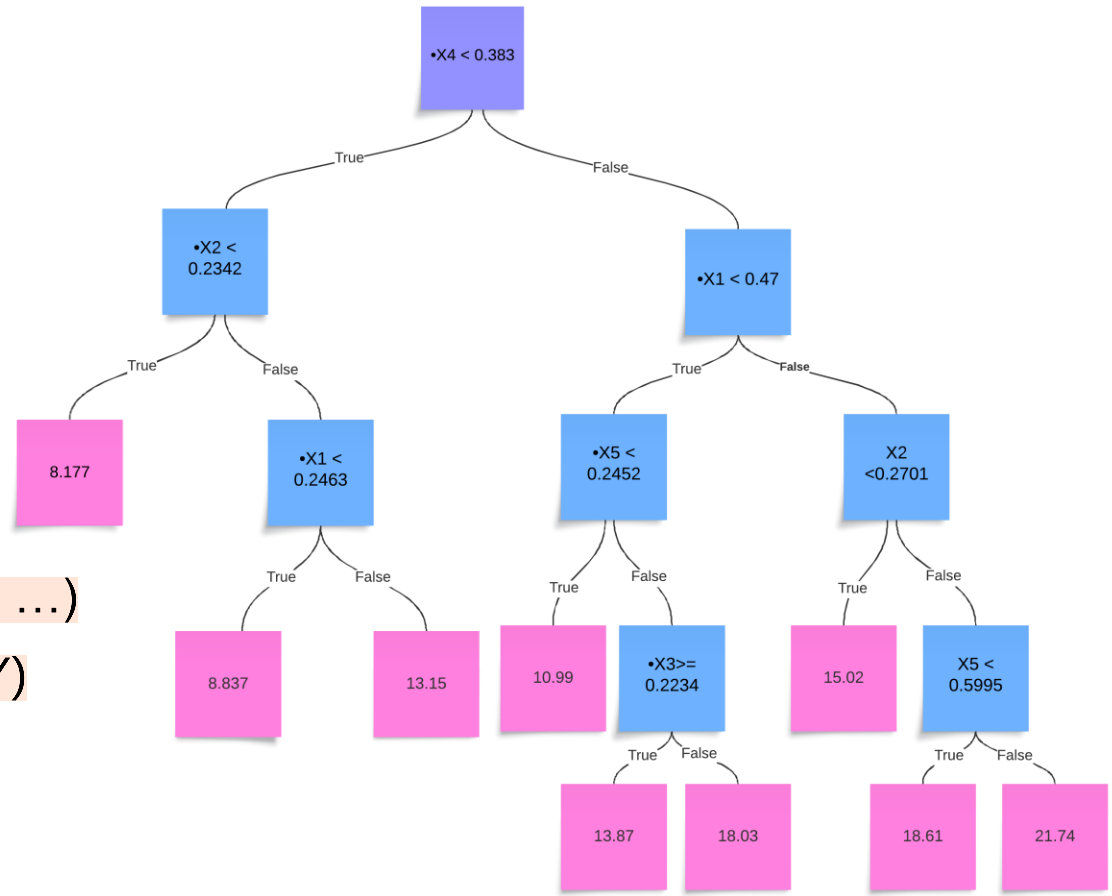
- AID: automatic interactive decision tree (Morgan and Sonquist, 1963)
  - High risk of overfitting → misleading conclusions
  - Lack of analytical rigor
- A group of computer scientists found similarities between DT and KNN
  - Terminal node trees → dynamical NN classifier (neighborhood)

Decision trees: terminology

- Root node
- (Decision) Node
- Terminal node / leaves

- Branch
- Split

- Attribute or features (X1, X2, …)
- Response/target variables (Y)

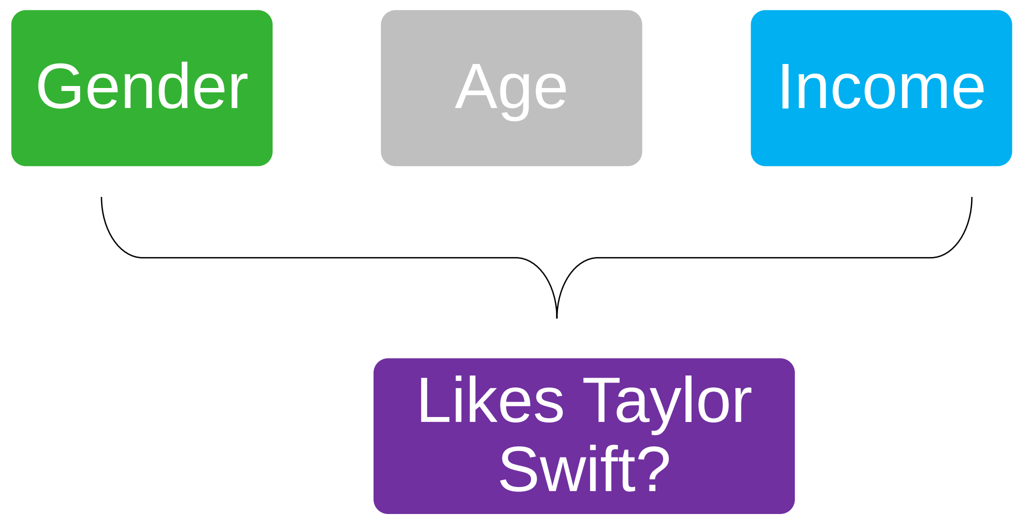


Example

We would like to know whether a person likes or not ***Taylor Swift*** considering ***age***, ***gender*** and ***monthly income***

df

	age	gender	monthly_income	likes_taylor_swift
0	53	1	1199	1
1	43	1	1475	1
2	29	1	2306	0
3	57	1	689	1
4	22	1	1457	1
5	35	0	1186	1
6	53	0	1457	1
7	33	1	1062	0
8	37	1	2399	1
9	25	1	2090	0
10	25	0	1767	1
11	38	1	1331	1
12	50	0	2028	1
13	54	0	1654	0
14	38	0	2008	1
15	17	0	2342	0
16	36	0	1146	1
17	16	1	520	0
18	38	1	1340	1
19	58	1	666	0

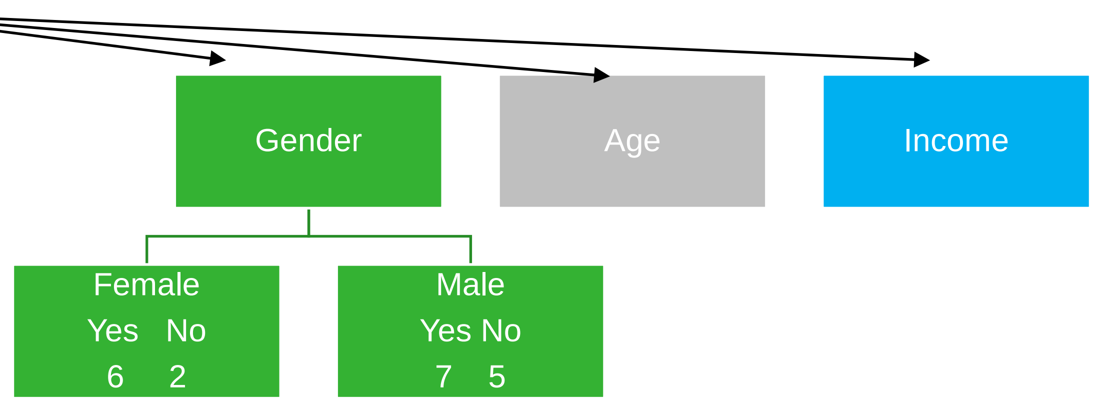


Example

- Create a root node
- Whether ***Gender, Age*** or ***Income*** be the question?
- Split condition, terminal node?, label

df

	age	gender	monthly_income	likes_taylor_swift
0	53	1	1199	1
1	43	1	1475	1
2	29	1	2306	0
3	57	1	689	1
4	22	1	1457	1
5	35	0	1186	1
6	53	0	1457	1
7	33	1	1062	0
8	37	1	2399	1
9	25	1	2090	0
10	25	0	1767	1
11	38	1	1331	1
12	50	0	2028	1
13	54	0	1654	0
14	38	0	2008	1
15	17	0	2342	0
16	36	0	1146	1
17	16	1	520	0
18	38	1	1340	1
19	58	1	666	0



Impure node: how to quantify ***impurity***?
Gini coefficient, entropy, log_loss

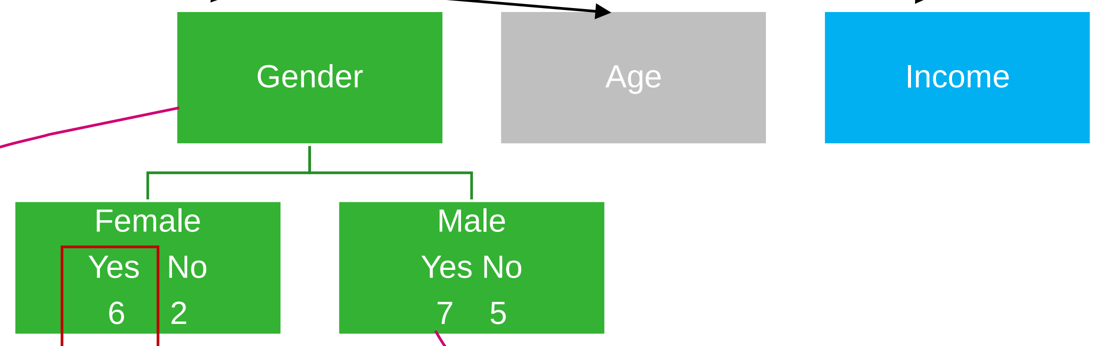
Example

- Gini impurity

df

	age	gender	monthly_income	likes_taylor_swift
0	53	1	1199	1
1	43	1	1475	1
2	29	1	2306	0
3	57	1	689	1
4	22	1	1457	1
5	35	0	1186	1
6	53	0	1457	1
7	33	1	1062	0
8	37	1	2399	1
9	25	1	2090	0
10	25	0	1767	1
11	38	1	1331	1
12	50	0	2028	1
13	54	0	1654	0
14	38	0	2008	1
15	17	0	2342	0
16	36	0	1146	1
17	16	1	520	0
18	38	1	1340	1
19	58	1	666	0

Training set



Gini impurity for a leaf = 1 – (probability of "Yes")^2 - (probability of "No")^2

$$\text{Gini} = 1 - \left(\frac{6}{6+2}\right)^2 - \left(\frac{2}{6+2}\right)^2 = 0.375$$

$$\text{Gini} = 1 - \left(\frac{7}{7+5}\right)^2 - \left(\frac{5}{7+5}\right)^2 = 0.486$$

$$\text{Weighted Gini} = \frac{8}{20}(0.375) + \frac{12}{20}(0.486) = 0.4416$$

Example

- Splitting numeric variables
- Sorting
- Average
- Gini coefficient

Gender | Age < 16.5 | Income

Yes	No
0	1

Yes	No
13	6

16.5
19.5
23.5
25.0
27.0
31.0

age	gender	monthly_income	likes_taylor_swift
16	male	520	0
17	female	2342	0
22	male	1457	1
25	male	2090	0
25	female	1767	1
29	male	2306	0
33	male	1062	0

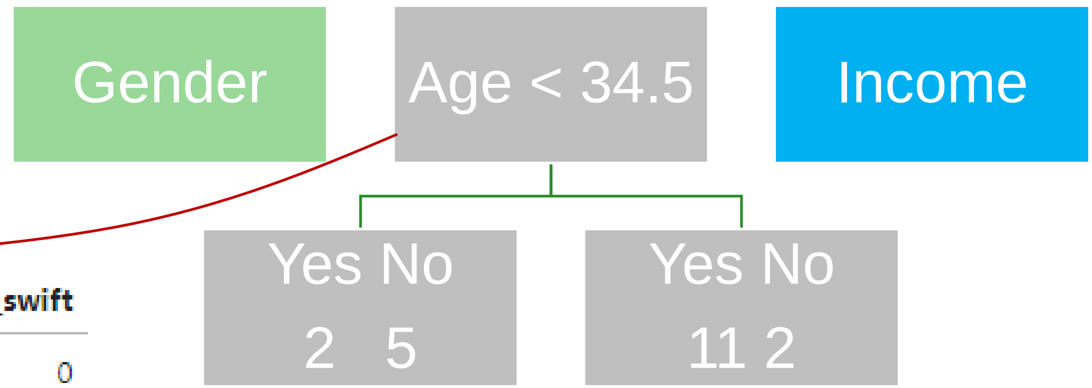
$$\text{Gini} = 1 - \frac{0}{0+1}^{2} - \frac{1}{0+1}^{2} = 0 \longrightarrow \text{Pure node}$$

$$\text{Gini} = 1 - \frac{13}{13+6}^{2} - \frac{6}{13+6}^{2} = 0.432$$

$$\text{Weighted Gini} = \frac{1}{20}(0) + \frac{12}{20}(0.432) = 0.410$$

Example

- Splitting numeric variables
- Sorting
- Average
- Gini coefficient



age	gender	monthly_income	likes_taylor_swift
16	male	520	0
17	female	2342	0
22	male	1457	1
25	male	2090	0
25	female	1767	1
29	male	2306	0
33	male	1062	0
35	female	1186	1
36	female	1146	1

16.5
19.5
23.5
25.0
27.0
31.0
$\boxed{34}$
35.5

$$\text{Gini} = 1 - \left(\frac{2}{2+5}\right)^2 - \left(\frac{5}{2+7}\right)^2 = 0.408$$

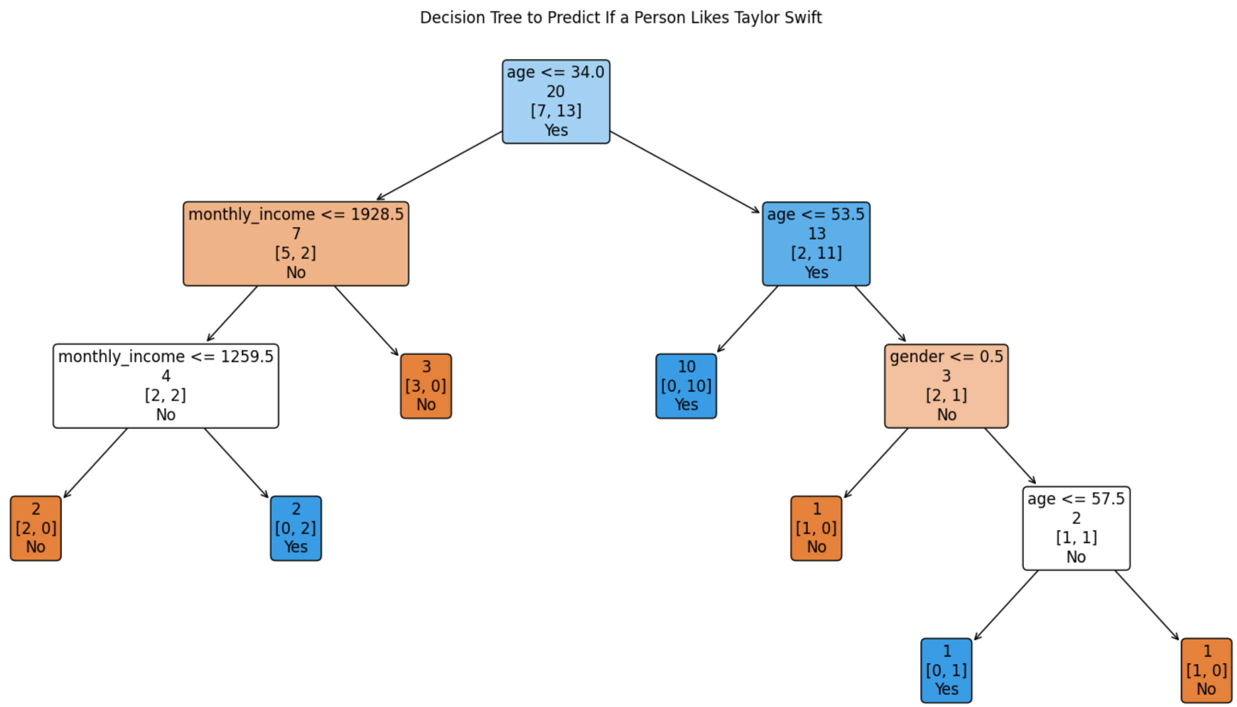
$$\text{Gini} = 1 - \left(\frac{11}{11+2}\right)^2 - \left(\frac{2}{11+2}\right)^2 = 0.26$$

$$\text{Weighted Gini} = \frac{7}{20}(0) + \frac{13}{20}(0.432) = 0.311$$

What is a decision tree?

df

	age	gender	monthly_income	likes_taylor_swift
0	53	1	1199	1
1	43	1	1475	1
2	29	1	2306	0
3	57	1	689	1
4	22	1	1457	1
5	35	0	1186	1
6	53	0	1457	1
7	33	1	1062	0
8	37	1	2399	1
9	25	1	2090	0
10	25	0	1767	1
11	38	1	1331	1
12	50	0	2028	1
13	54	0	1654	0
14	38	0	2008	1
15	17	0	2342	0
16	36	0	1146	1
17	16	1	520	0
18	38	1	1340	1
19	58	1	666	0



Overfitting

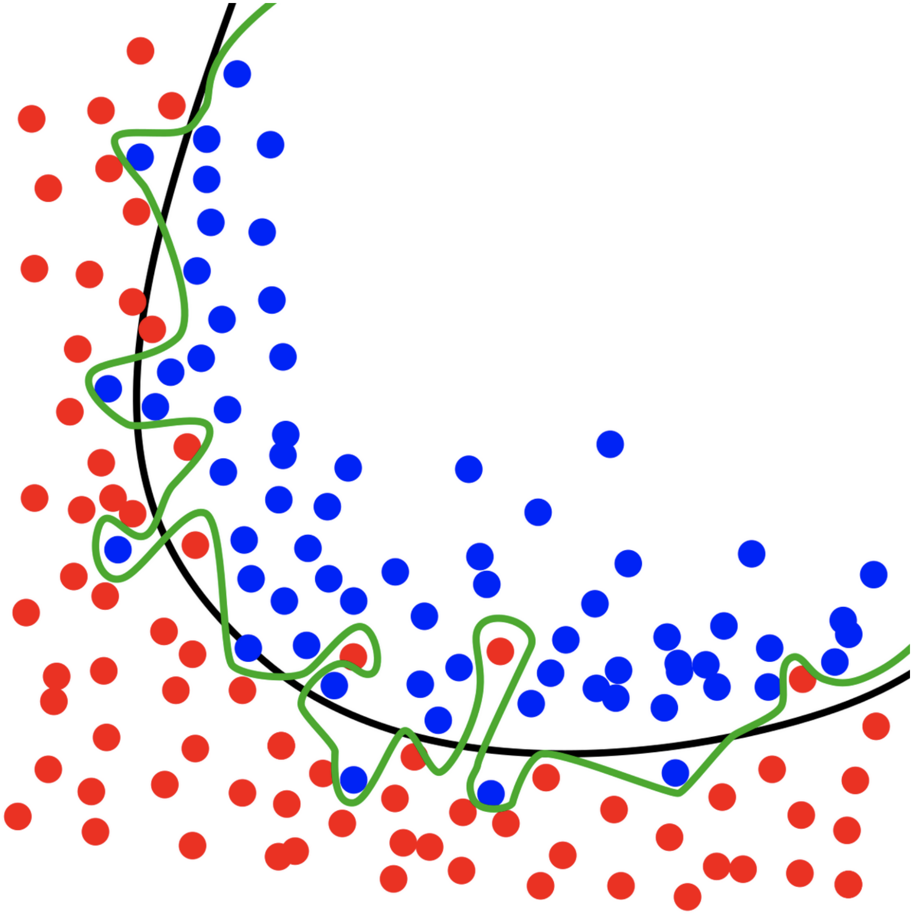


Figure 1. The green line represents an overfitted model and the black line represents a regularized model. While the green line best follows the training data, it is too dependent on that data and it is likely to have a higher error rate on new unseen data, compared to the black line.

Decision trees: CART

These group of scientists led by Leo Breiman (1928-2005) invented:

- CART: Classification and Regression Trees (Breiman et al., 1984)

- CART is one of the most popular DT methods because
  - It can cope with continuous and categorical data (both as targets and as predictors)
  - It is analytically rigorous

NOTE: CART is also a commercial software package (Salford Systems)

CART algorithm

- Start by creating the root node (all data)
- Root → 2 children → 4 grandchildren….
- “Grow” the tree until no further splits are possible (lack of data).
- “Prune” back using the cost-complexity method.
  - Splits are pruned sequentially according to their contribution to the performance on the training data.
  - Remove less relevant splits
- Evaluate the set of nested pruned trees by using an independent dataset
  - Or use cross-validation

CART algorithm (II)

- The use of DT require a clear definition of the following 3 elements:

1. A way to select a split at every intermediate node
2. A rule for determining if a node is a terminal one
3. A rule for assigning a value (Y_{est}) to each terminal node

(Plant, 2012)

Splitting rules (intermediate nodes)

- An object goes left IF the chosen attribute meets some CONDITION, otherwise it goes right
  - Continuous data: X <= Condition
  - Nominal data: X belongs to set {A,B,C,D}

- The splitter and the split point are chosen by CART
  - Always binary splits
  - An attribute can be used multiple times

Rule node is terminal

- CART grows the tree until all the data is the resulting node is homogeneous or it contains less elements than a (chosen) threshold

- After a maximum tree has been created, it is pruned back using bias-variance trade off
  - Bias (e.g., MSE) and Variance (~ number of end nodes).
  - Use of cross-validation to minimize the Bias + Variance

Value terminal node

- For categorical data
  - Y_{est} (t)= Mode of the labels of all the elements in the terminal node

- For continuous data
  - Y_{est} (t)= $\frac{1}{n(t)} \sum_{Xi \in t} Y_i$

  So… the mean value of the response variable in the terminal node

Decision Trees

- Advantages
  - Output is easy to understand
  - Can combine numeric and categorical data
  - Robust (outliers)
  - Fast (after developing the rules)
- Disadvantages
  - Overfitting
  - Limited to the range of the attributes in the training data
  - Unstable (small perturbation input → larger perturbation output)
    - Categorical data?

Random forests

- Leo Breiman continued working on DT and around the year 2000 he found and demonstrated that regression results and classification accuracy can be improved by using ensembles of trees where each tree grown in a "random" fashion.

- This work resulted in "random forests"

- Ensemble = a set of elements.
- Ensemble methods are becoming highly popular → computer power

Random forests (II)

- RF are fast and easy to implement.
- They yield highly accurate predictions
  (even if the input data has a high dimensionality)
- No overfitting
- Provide insight on the importance of each attribute/feature/dimension
- They are easily parallelizable
- Data does not need pre-processing

- They are one of the most popular general-purpose ML methods

Random forests (III)

- RF generates an ensemble of decision trees during the training
- Each tree is the result of applying the CART method to a random selection of attributes/features at each node.
- And of using a random subset of the original input data (chosen with replacement, -- bootstrapping || Bagging = bootstrapping aggregation)
- Response variables are obtained by voting/averaging over the ensemble

Random Forest: algorithm

- Input data: N training cases each with M variables

- n out of N samples are chosen with replacement (bootstrapping).
- Rest of the samples to estimate the error of the tree (out of bag)
- m << M variables are used to determine the decision at a node of the tree
- Each tree is fully grown and not pruned

- Output of the ensemble: aggregation of the outputs of the trees

Random Forest

- Bagging → boostrap aggregation

	age	gender	monthly_income	likes_taylor_swift
0	53	male	1199	1
1	43	male	1475	1
2	29	male	2306	0
3	57	male	689	1
4	22	male	1457	1
5	35	female	1186	1
6	53	female	1457	1
7	33	male	1062	0
8	37	male	2399	1
9	25	male	2090	0
10	25	female	1767	1
11	38	male	1331	1
12	50	female	2028	1
13	54	female	1654	0
14	38	female	2008	1
15	17	female	2342	0
16	36	female	1146	1
17	16	male	520	0
18	38	male	1340	1
19	58	male	666	0

	age	gender	likes_taylor_swift
0	53	male	1
1	43	male	1
2	29	male	0
3	57	male	1
4	22	male	1
5	35	female	1
1	43	male	1
7	33	male	0
8	37	male	1
9	25	male	0
10	25	female	1

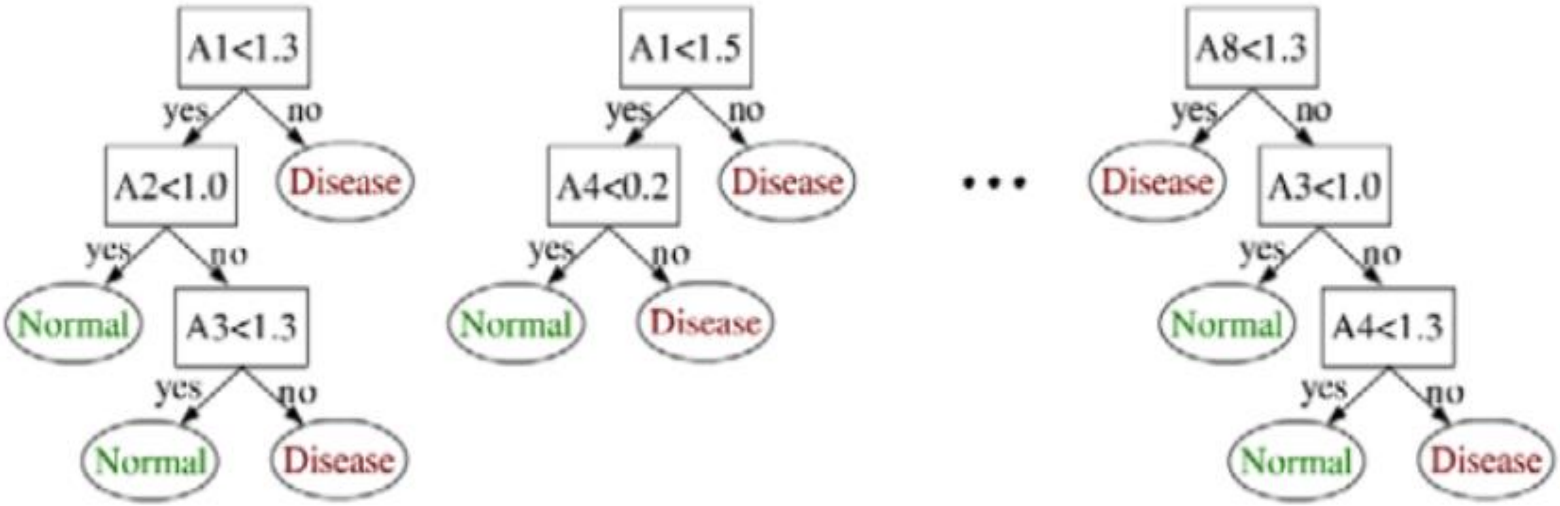
Training subset 1

...

	age	monthly_income	likes_taylor_swift
7	33	1062	0
8	37	2399	1
9	25	2090	0
10	25	1767	1
11	38	1331	1
12	50	2028	1
13	54	1654	0
14	38	2008	1
15	17	2342	0
16	36	1146	1

Training subset n

Random Forest: an example



Random Forest

- Advantages
  - No pruning needed
  - High Accuracy
  - Provides variable importance
  - No overfitting || Not very sensitive to outliers

- Disadvantages
  - Cannot predict (regression) beyond range of input parameters
  - Smoothing extreme values (underestimate high values; overestimate low values)
  - More difficult to visualize/interpret

Spatial and temporal data ?!

- DT (and RF) do not directly use spatio-temporal information
- They only make use of the attributes/values at all the sampled locations and times

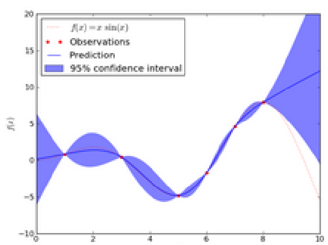
- Remember to always examine the spatial variability of the results to check the “validity” of the classification and/or regression.
- Do not forget to make use of maps and other geovisualizations

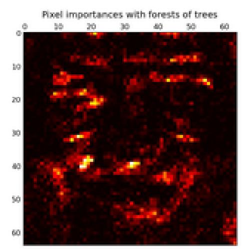
DT & RF software

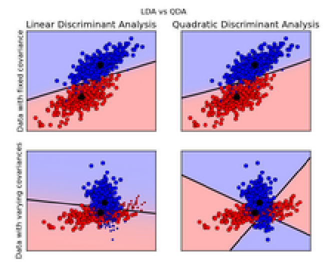
- R packages
  - Party
  - Rpart
  - Randomforest
  - …
- Python
  - Scikits learn (sklearn)
  - …

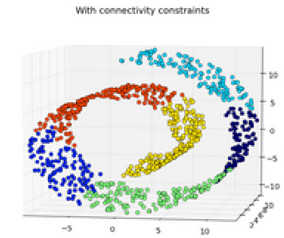
scikits learn

machine learning in Python









Easy-to-use and general-purpose machine learning in Python

Scikit-learn integrates **machine learning** algorithms in the tightly-knit scientific **Python** world, building upon numpy, scipy, and matplotlib. As a machine-learning module, it provides versatile tools for data mining and analysis in any field of science and engineering. It strives to be **simple and efficient**, accessible to everybody, and reusable in various contexts.

Supervised learning

Support vector machines, linear models, naive Bayes, Gaussian processes...

Unsupervised learning

Clustering, Gaussian mixture models, manifold learning, matrix factorization, covariance...

And much more

*Model selection, datasets, feature extraction... **See below**.*

Spatio-temporal analytics and modeling

A decisive tree

Questions??