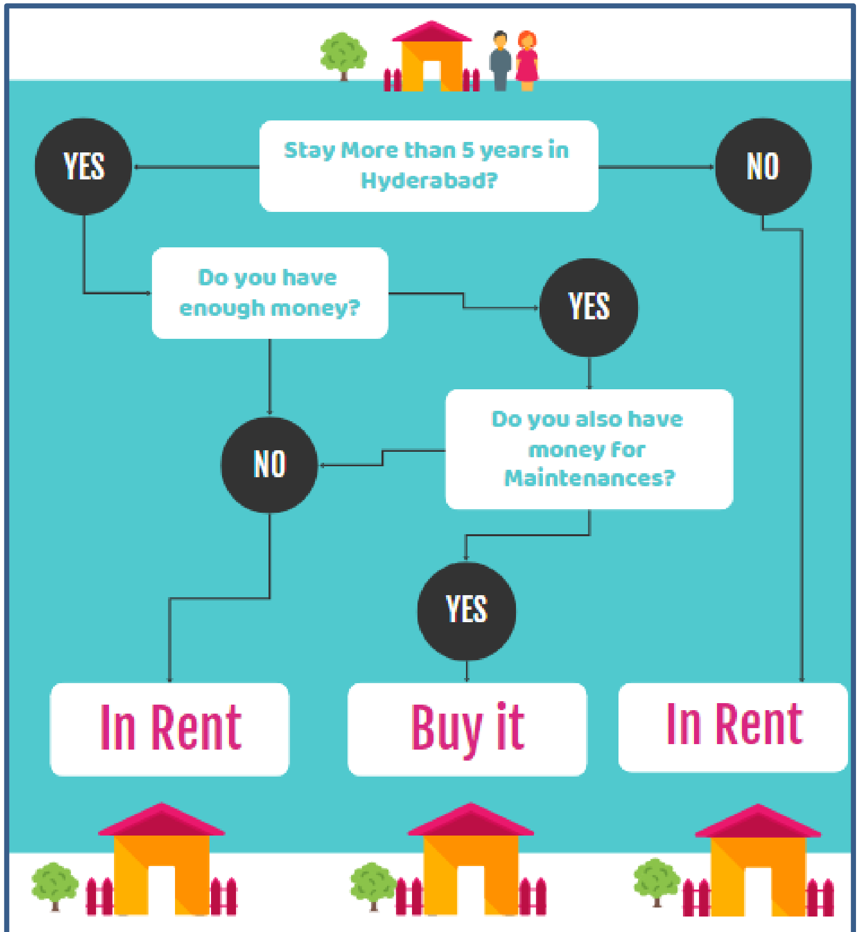
28.08.2024

BITS F464: Machine Learning (1st Sem 2024-25)

SUPERVISED LEARNING-II: DECISION TREES AND ENSEMBLE LEARNING

Chittaranjan Hota, Sr. Professor
Dept. of Computer Sc. and Information Systems
hota@hyderabad.bits-pilani.ac.in

Decision Tree: Applications



ENCODING

```
If (stay > 5yrs)
{
   If (money==80L)
   {
      If (m_money == 10L)
      {
         Buy a house;
      }
       Stay in rent;
   }
   Stay in rent;
}
Stay in rent;
```

- Equipment classification, Medical diagnosis, Credit Risk analysis, …

Applications continued…

XBox

KINECT SPORTS

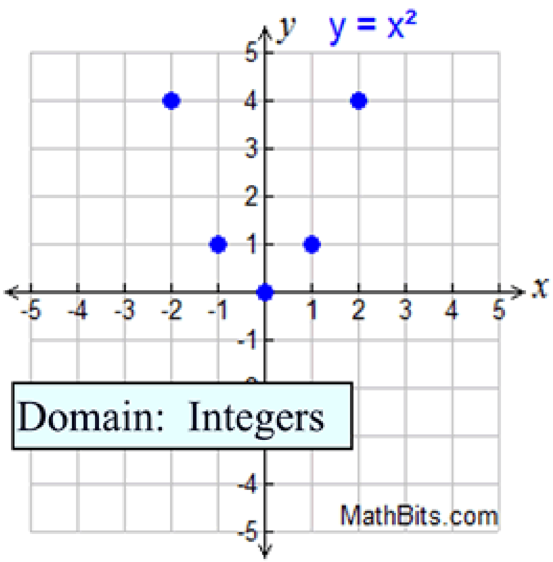
Real-Time Human Pose Recognition in Parts from Single Depth Images

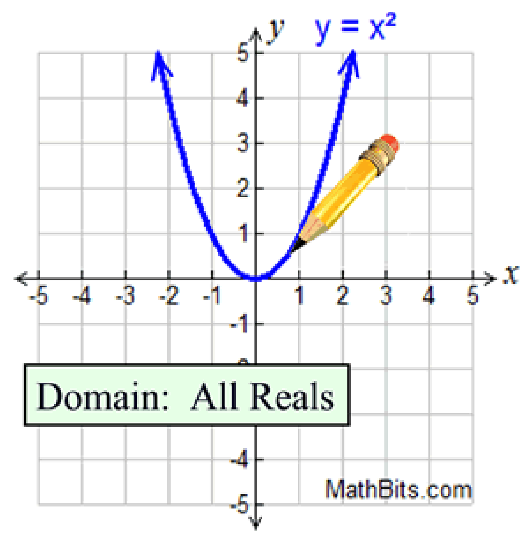
Jamie Shotton Andrew Fitzgibbon Mat Cook Toby Sharp Mark Finocchio
Richard Moore Alex Kipman Andrew Blake
Microsoft Research Cambridge & Xbox Incubation

HANSON ROBOTICS

What is Decision Tree Learning

- Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree.





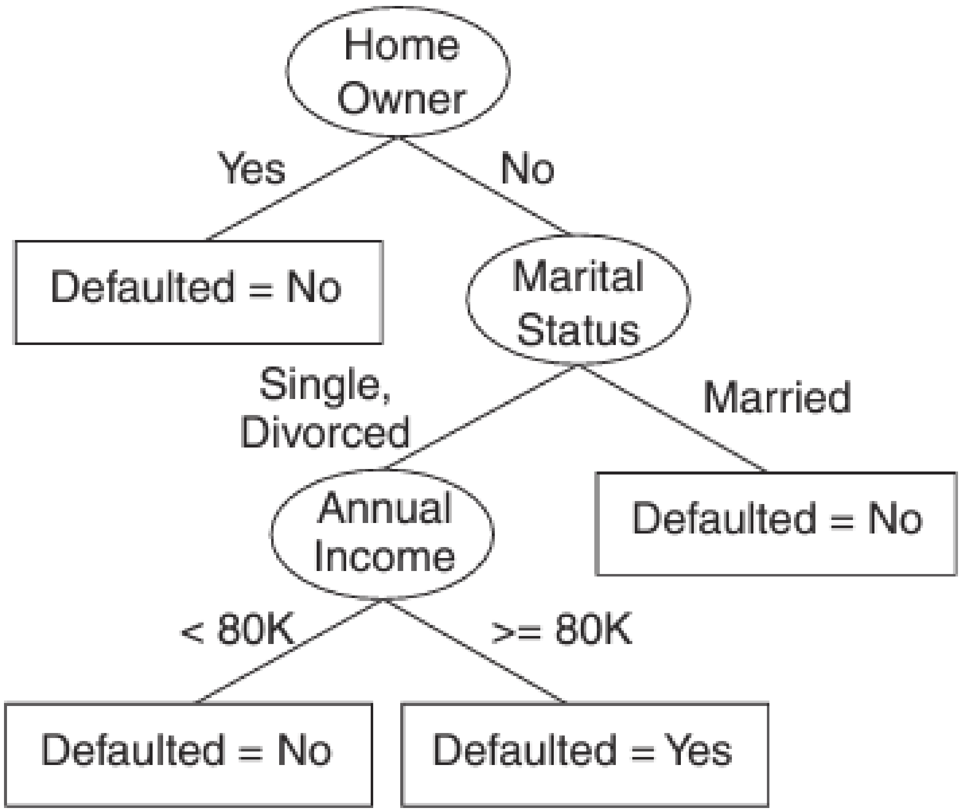
continuous

- No. of students present in the class.
- No. of holidays in this sem.
- No. of courses you finish.

- Height of a person.
- Temperature in this room.
- USD value in rupees.

Decision Tree: Learning by Induction

	binary	categorical	continuous	class
Tid	**Home Owner**	**Marital Status**	**Annual Income**	**Defaulted Borrower**
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



(d)

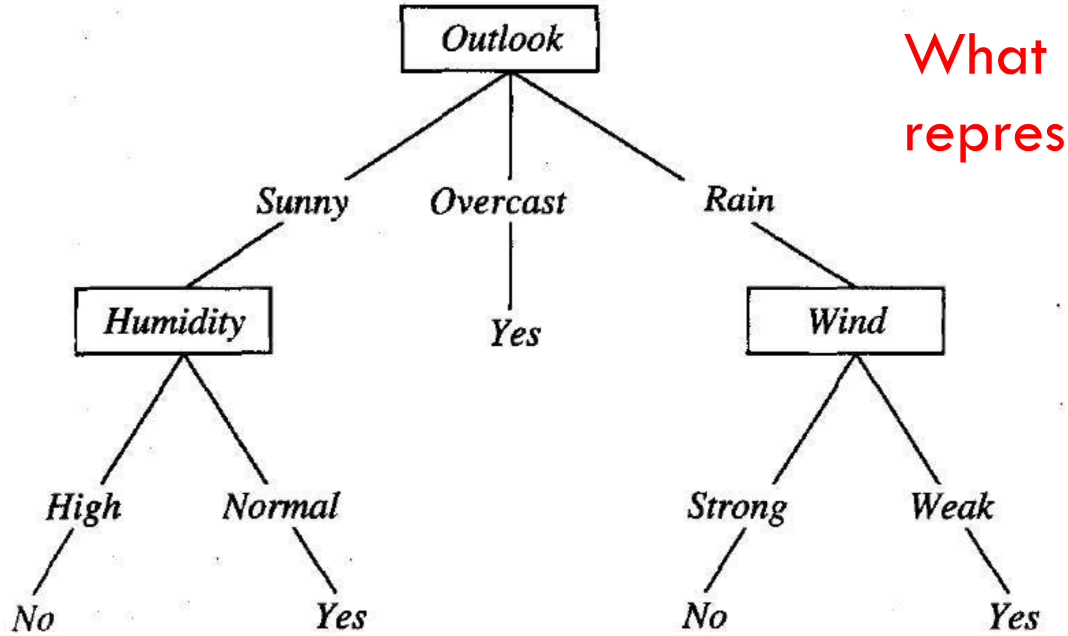
Continued…

- A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks.
- Classification: Playing Tennis, loan defaulter; Regression: How many students will enroll into ML next semester, What will be the cost of Honda Amaze next year?
- Parametric Vs Non-parametric models
  - A model learning from the data assuming a fixed number of parameters Vs. No-assumption or no prior-knowledge about data distribution (free to learn any functional form from data).
  - Linear/Logistic regression (coefficients), perceptron (weights) Vs. SVM, DT, KNN, Complex NNs.
  - Fast, Simple and Less data Vs. Slower, Complex and More data.

Decision Tree Representation

- Hypothesis space is disjunction of conjunctions, while candidate-elimination (version space) algorithm could only accommodate what? Given a test input:

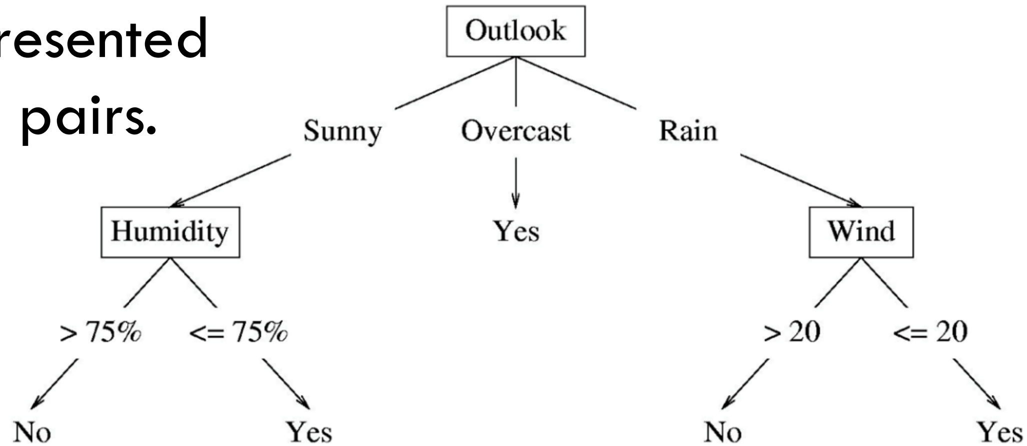
$$\langle Outlook = Sunny,\ Temperature = Hot,\ Humidity = High,\ Wind = Strong \rangle$$



What is the expression that represents this decision tree?

Problem Characteristics

- Instances are represented by attribute-value pairs.

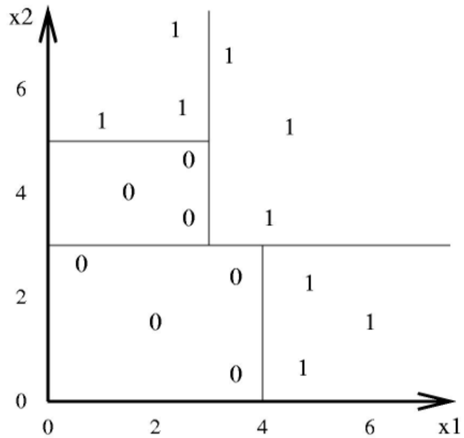


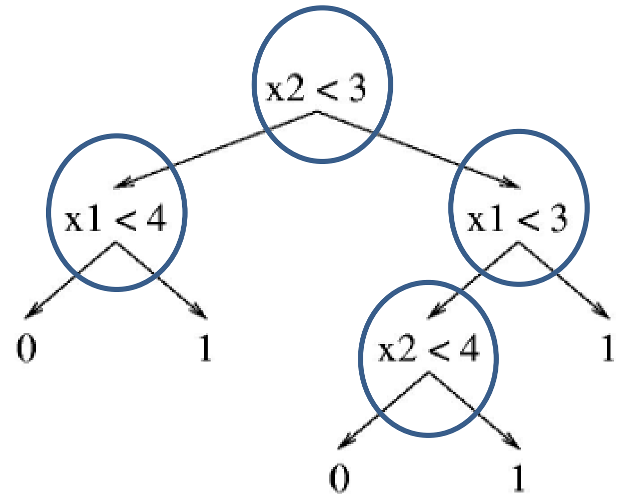
- In majority of the cases the target function has discrete output values. Either 2 or more output values are possible. However, real-valued outputs are also possible.
- Robust to errors: classification and attribute value errors.
- Training data may contain missing attribute values.

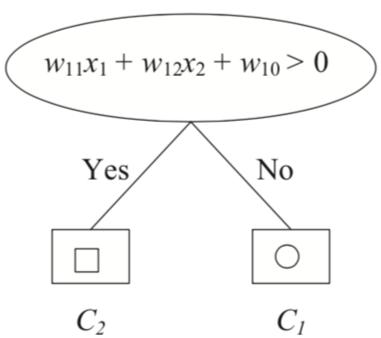
Decision Boundary in Decision Trees

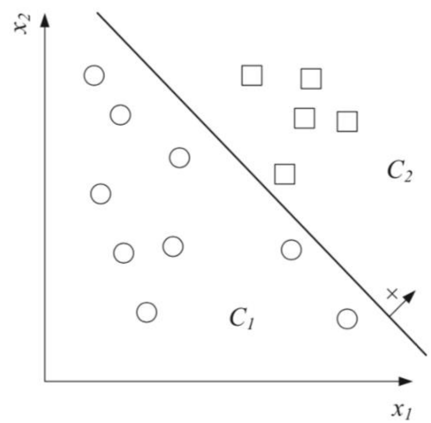
- Decision Trees divide the feature space into axis-parallel rectangles, and label each rectangle with one of the K-classes.

Univariate Tree:
In each internal node tree uses only one dimension or attribute. (ID3)









In a *Multivariate tree*, at a decision node, all input dimensions can be used and thus it is more general. (CART)

Decision Tree Learning Algorithm

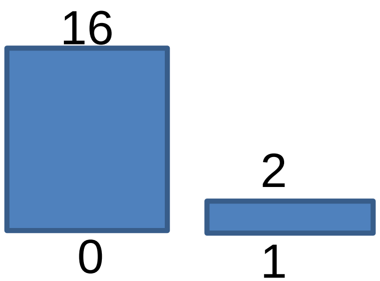
```
TreeGrowth (E, F)
if stopping_cond(E,F) == true then
    leaf = createNode().
    leaf.label = Classify(E).
    return leaf.
else
    root = createNode().
    root.test_cond = find_best_split(E, F).
    let V = {v|v is a possible outcome of root.test_cond }.
    for each v ∈ V do
        E_v = {e | root.test_cond(e) = v and e ∈ E}.
        child = TreeGrowth(E_v, F).
        add child as descendent of root and label the edge (root → child) as v.
    end for
end if
return root.
```

Quantifying Uncertainty

What is Entropy? $\mathrm{H}(X) := -\sum_{x\in\mathcal{X}} p(x)\log p(x)$

Sequence 1:

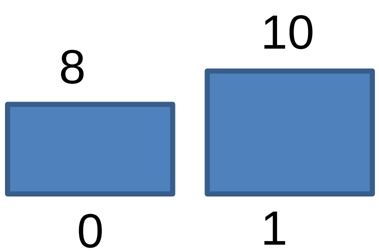
0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0



$$-\frac{16}{18}\log_2\frac{16}{18} - \frac{2}{18}\log_2\frac{2}{18} = 0.503$$

Sequence 2:

1 0 0 1 0 1 1 1 0 1 0 1 1 0 0 1 0 1

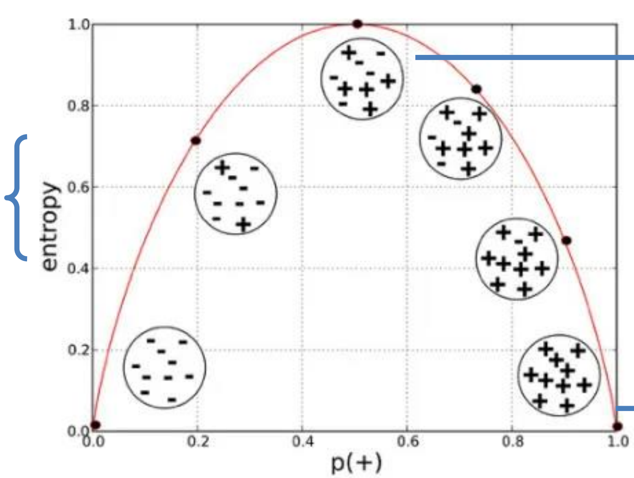


$$-\frac{8}{18}\log_2\frac{8}{18} - \frac{10}{18}\log_2\frac{10}{18} = 0.991$$

Optimal length code assigns $-\log_2 p$ bits to message having probability p.

Entropy Continued...

- Deterministic: good(all are true or false; one class in the leaf)
- Uniform distribution: bad (all classes in leaf equally probable)
- What about distributions in between?
- Entropy in information theory specifies the minimum number of bits needed to encode the class code of an instance.



(Let S be a collection)

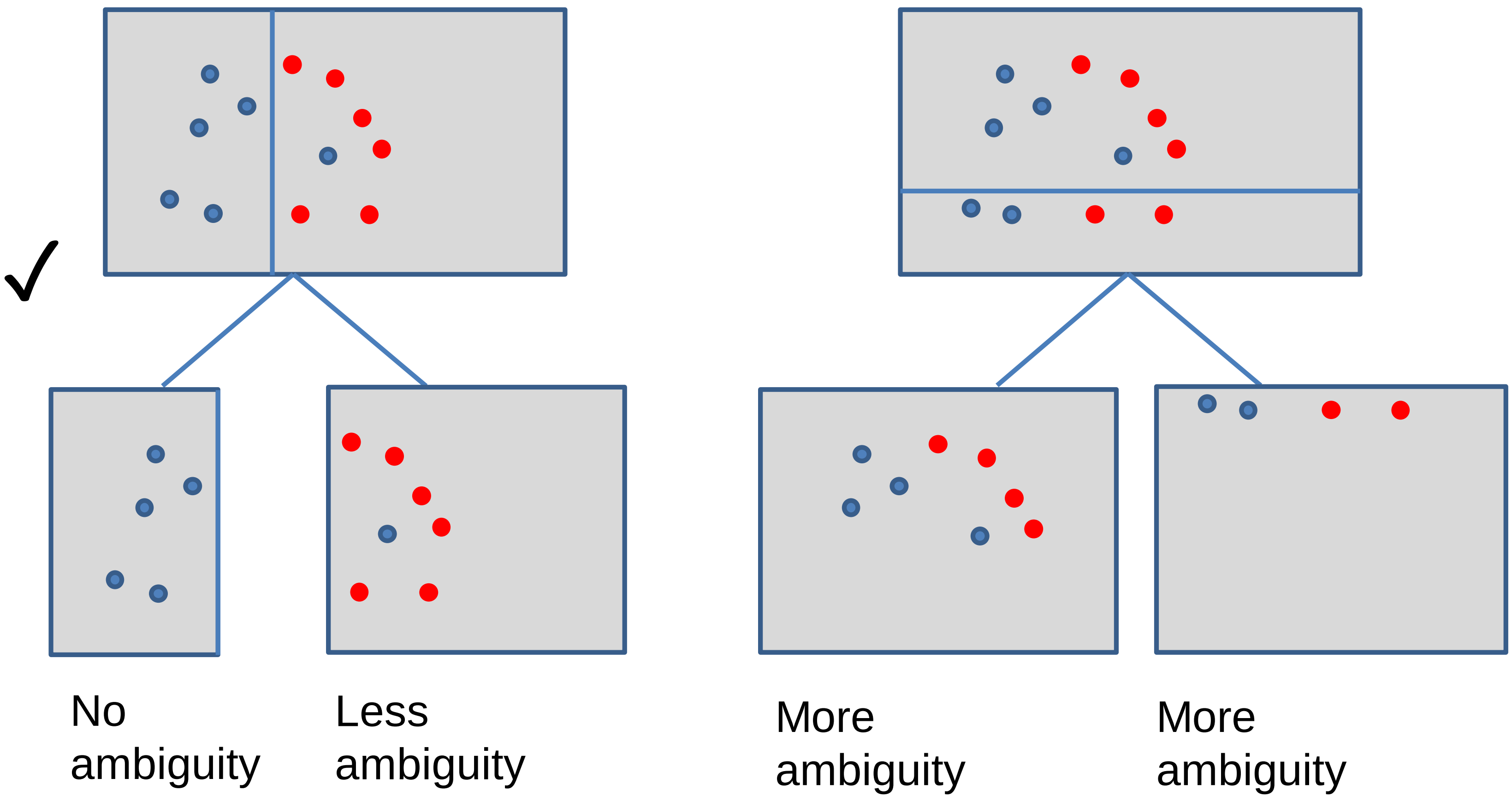
Equal no. of + and - examples, what is H?

Unequal no. of +ve and -ve, what is H?

All members of S belong to the same class, what is H?

Choosing Attribute/value at each level

Which one is better?



Information Gain

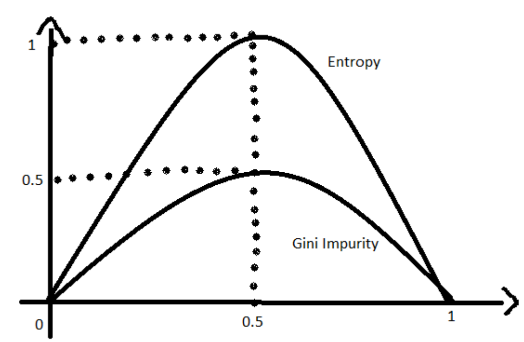
- Measures how well an attribute divides the training examples according to their target types.
- Decline in Entropy.

ID-3: Iterative Dichotomiser 3

$$Information\ Gain\ (S, A) = Entropy\ (S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- Top down greedy heuristics: Ross Quinlan

$$\text{Gini Index} = 1 - \sum_{i=1}^{n} (P_i)^2$$



Gini Impurity:

- Is a metric to measure how often a randomly chosen element would be incorrectly identified. Attribute with lower G.I should be preferred. CART

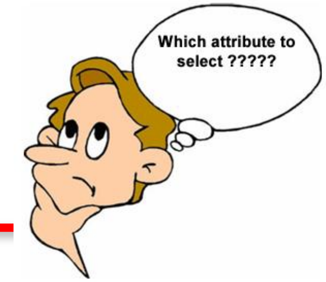
- Gini Index is easier to compute and is more focused on purity.
- Information Gain involves entropy calculations and favors attributes that reduce uncertainty the most.

An Example: Play Tennis

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Entropy of the Training Set: $E(S) = E([9+,5-]) = (-9/14 \log_2 9/14) + (-5/14 \log_2 5/14) = 0.94$

Continued…



The information gain for Outlook is:

- I.G(S, Outlook) = E(S) – [5/14 * E(Outlook=sunny) + 4/14 * E(Outlook = overcast) + 5/14 * E(Outlook=rain)]
- I.G(S, Outlook) = E([9+,5-]) – [5/14 * E(2+,3-) + 4/14 * E([4+,0-]) + 5/14 * E([3+,2-])]
- I.G(S, Outlook) = 0.94 – [5/14 * 0.971 + 4/14 *0.0 + 5/14*0.971]
- I.G(S, Outlook) = 0.246

The information gain for Temperature is:

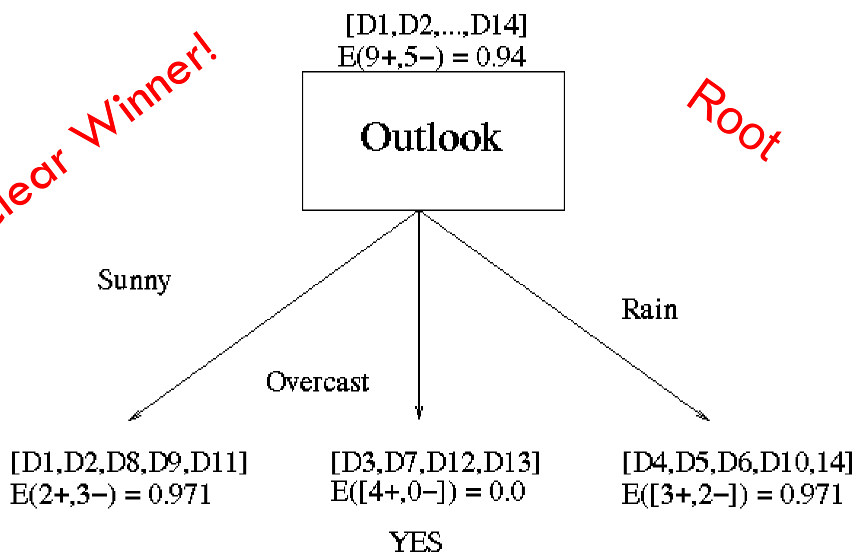
- I.G(S, Temperature) = 0.94 – [4/14 * E(Temperature=hot) + 6/14 * E(Temperature=mild) + 4/14 * E(Temperature=cool)]
- I.G(S, Temperature) = 0.94 – [4/14 * E([2+,2-]) + 6/14 * E([4+,2-]) + 4/14 * E([3+,1-])]
- I.G(S, Temperature) = 0.94 – [4/14 + 6/14*0.918 + 4/14*0.811]
- I.G(S, Temperature) = 0.029

Continued...

The information gain for Humidity is:

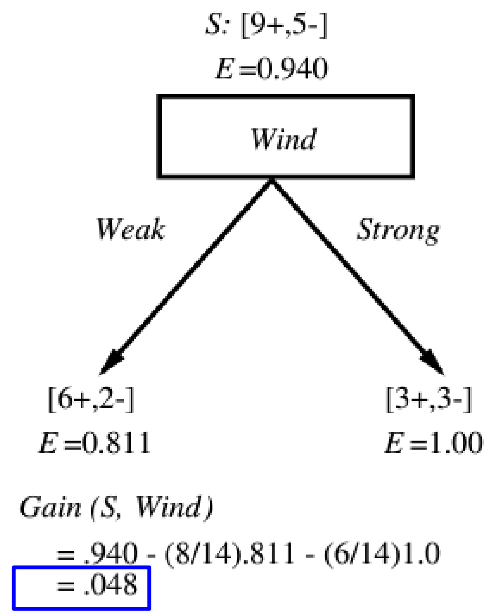
- I.G(S, Humidity) = 0.94 – [7/14 * E(Humidity=high) + 7/14 * E(Humidity=normal)]
- I.G(S, Humidity = 0.94 – [7/14 * E([3+,4-]) + 7/14 * E([6+,1-])]
- I.G(S, Humidity = 0.94 – [7/14 * 0.985 + 7/14 * 0.592]
- I.G(S, Humidity) = 0.1515

Clear Winner!



Root

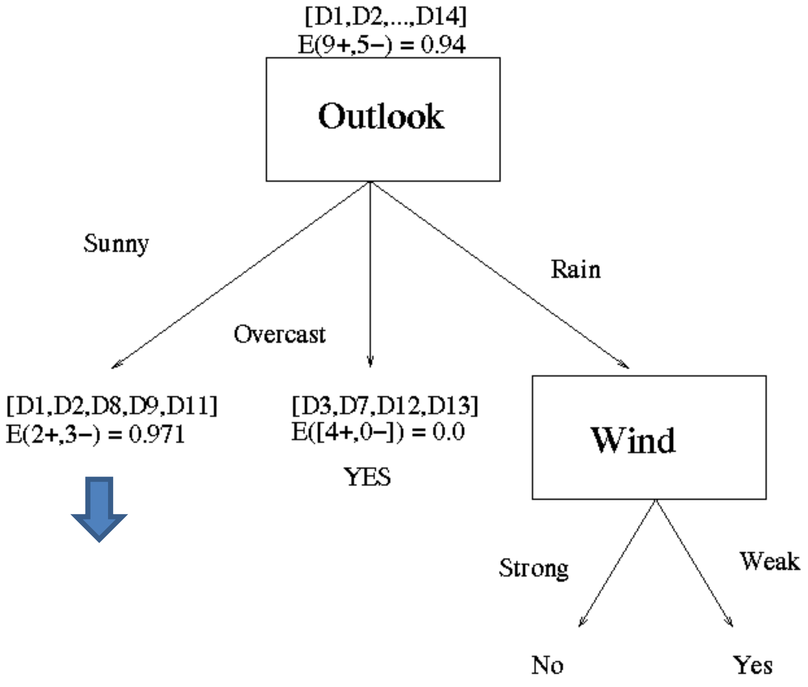
Information gain for Wind is:



Continued…

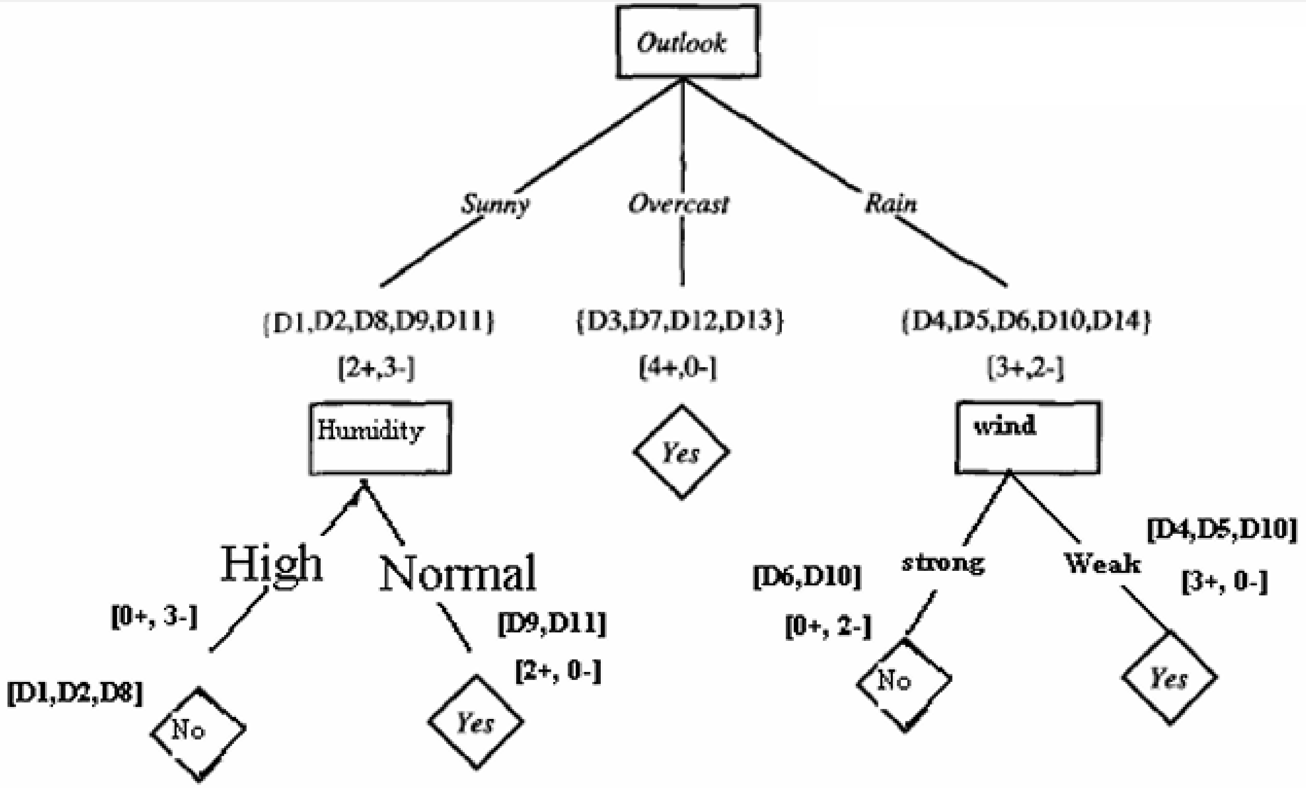
We should find out the nodes that will go below Sunny, Overcast, and Rain:

- I.G(Outlook = Rain, Humidity) = 0.971 – [2/5 * E (Outlook = Rain Λ Humidity = high) + 3/5 * E (Outlook = Rain Λ Humidity = normal]
- I.G(Outlook = Rain, Humidity) = 0.02
- I.G(Outlook = Rain, Wind) = 0.971- [3/5*0 + 2/5*0]
- I.G(Outlook = Rain, Wind) = <u>0.971</u>



Continued…

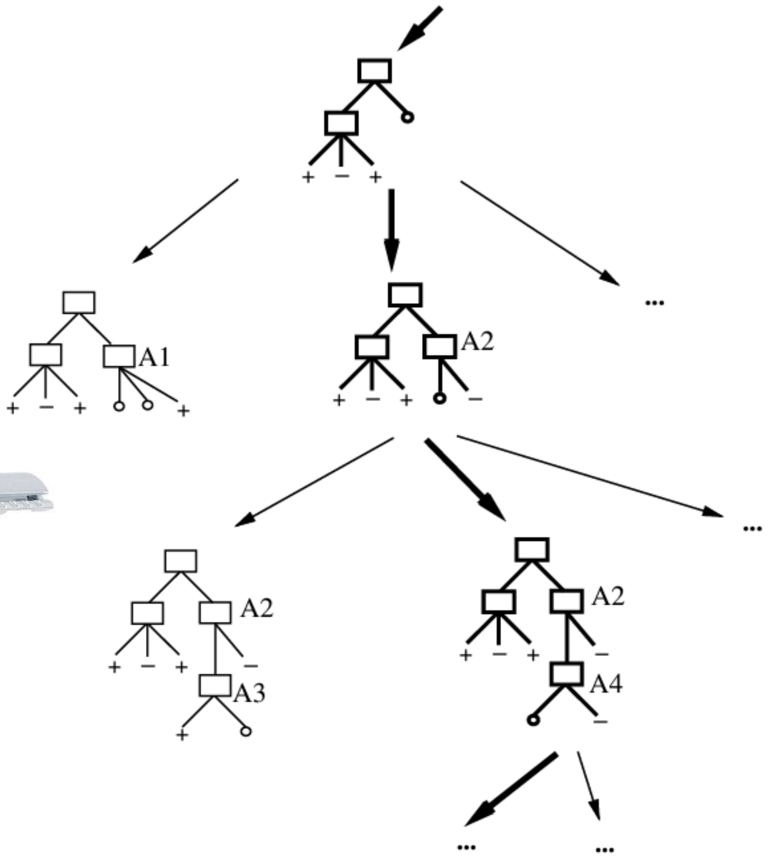
```
{'Outlook': {'Overcast': 'Yes',
             'Rain': {'Wind': {'Strong': 'No', 'Weak': 'Yes'}},
             'Sunny': {'Humidity': {'High': 'No', 'Normal': 'Yes'}}}}
```



(Assignment 2)

Hypothesis space search by ID3

- Selects trees that place the attributes with highest information gain closest to the root.
- Selects in favor of shorter trees over longer ones.
- A smaller (simpler) tree is more general than a larger (complex) tree.
- Hence, smaller one will be more accurate.
- This is what Occam's Razor says, i.e.



(Inductive Bias)

When two hypothesis equally explain the training set, pick the more general of the two.

What is the Idea here?

CART: Classification and Regression Trees

- DecisionTreeClassifier from Scikit Learn (Assignment 2)
- Example: Classify Apple and Orange based on Color and Weight features.

$$G = 1 - \sum_{i=1}^{m} p_i^2$$

4 Apples, 6 Oranges

G.I = 1 – (.4X.4 + .6X.6) = .48

Now, Suppose we choose Color attribute:

3 Apples, 2 Oranges

G.I = 1 – {$(3/5)^2 + (2/5)^2$} = .48

1 Apple, 4 Oranges

G.I = 1 – {$(1/5)^2 + (4/5)^2$} = .32

What is the Weighed Gini Index after the split?

$$Gini_{\text{split}} = \frac{N_1}{N} \times Gini_1 + \frac{N_2}{N} \times Gini_2$$

W.G.I = (5/10)X.48 + (5/10)X.32 = .40

What is the Interpretation here?

Now, find out for the weight attribute and see which one reduces G.I more.

C4.5: Gain Ratio

Definition: Gain Ratio is an extension of Information Gain that adjusts for the number of distinct values in an attribute, making it less biased toward attributes with many distinct values.

Calculation:

$$GR(T, A) = \frac{IG(T, A)}{SplitInfo(T, A)}$$

Where:

- $IG(T, A)$ is the Information Gain for attribute A.
- $SplitInfo(T, A)$ is the Split Information, calculated as:

$$SplitInfo(T, A) = -\sum_{v \in Values(A)} \frac{|T_v|}{|T|} \times \log_2 \left(\frac{|T_v|}{|T|} \right)$$

- $SplitInfo(T, A)$ measures the potential information generated by splitting the data on A.

C4.5 improves over ID-3 by handling both categorical and continuous attributes, dealing with missing values, and pruning trees after they are created to remove branches that reflect noise in the data.

ID-3, CART and C4.5

C4.5	Gain Ratio	Categorical and Numeric values	Pruning is used	Handles missing values.	Ross Quinlan, UNSW, UTS
ID3	Information Gain	Only Categorical value	No pruning	No	Ross Quinlan
CART	Gini Index	Categorical and Numeric values	Pruning is used	Handles missing values.	Leo Breiman, UC Berkeley

Overfitting

Target function is : $Y = X_1 \wedge X_2$

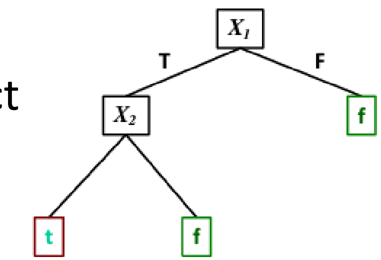
There is noise in some feature values.

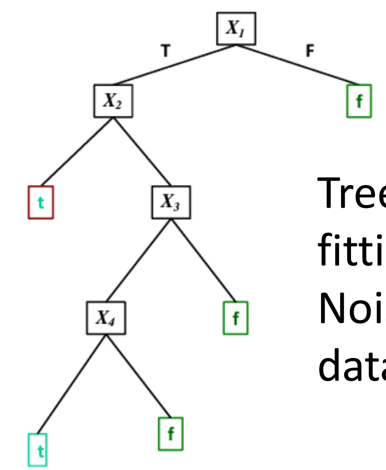
Training set:

X_1	X_2	X_3	X_4	X_5	…	Y
t	t	t	t	t	…	t
t	t	f	f	t	…	t
t	f	t	t	f	…	t
t	f	f	t	f	…	f
t	f	t	f	f	…	f
f	t	t	f	t	…	f

noisy value

Correct Tree



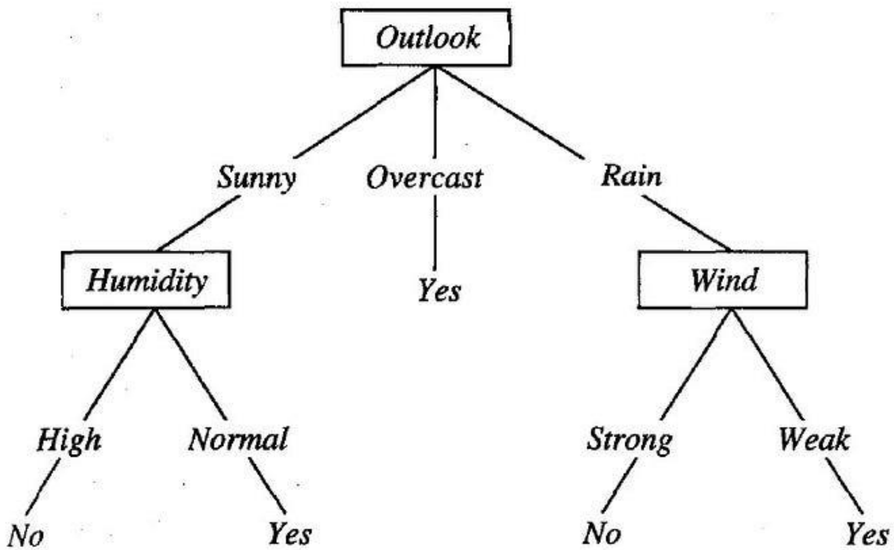


Tree fitting Noisy data

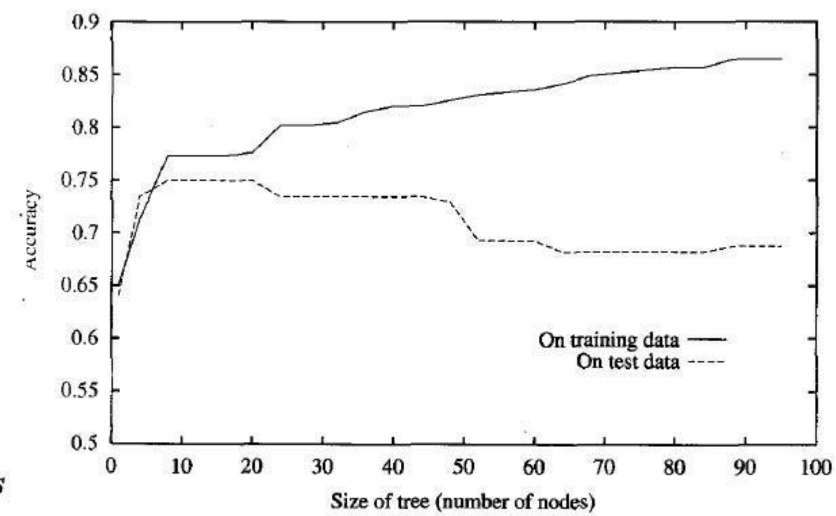
Source: Class slides of Ilias Diakonikolas, UW Madison

Overfitting in Decision Trees

- If the noisy sample D15 < Sunny, Hot, Normal, Strong, No > is incorrectly added to the previous set D1 to D14 [noisy because it would have been otherwise Yes (+ve)].



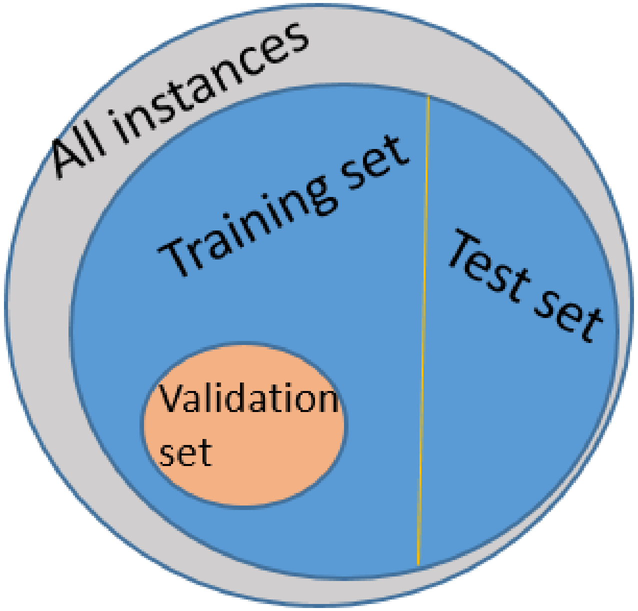
D9(+), D11(+),
D15(-)



Avoiding Overfitting

How can we avoid overfitting?

- Stop growing when data split
  significant
- Grow full tree, then post-prun



How to select "best" tree:

- Measure performance over trai
- Measure performance over separate validation data set
- Add complexity penalty to performance measure

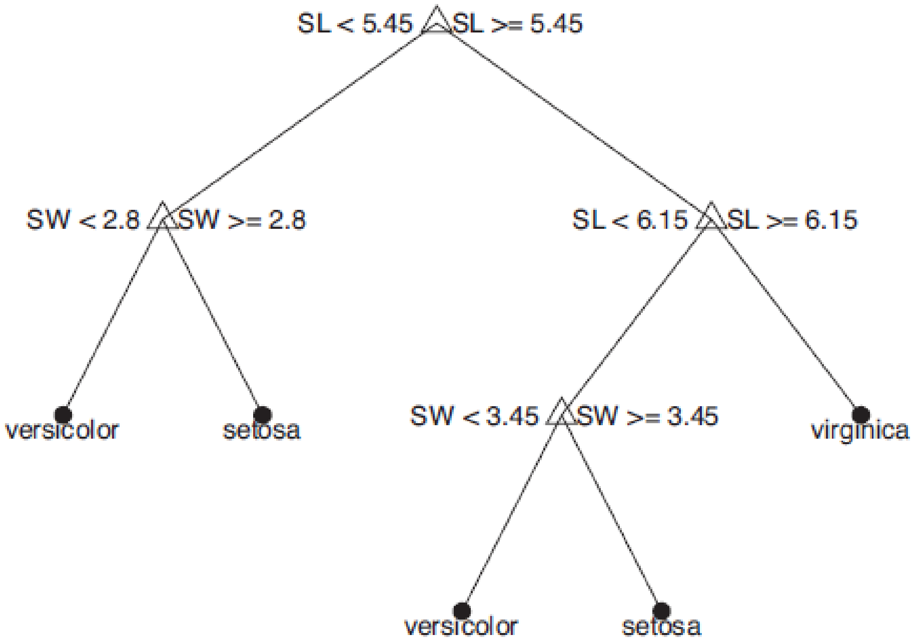
Reduced Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

CART Pruning Ex.



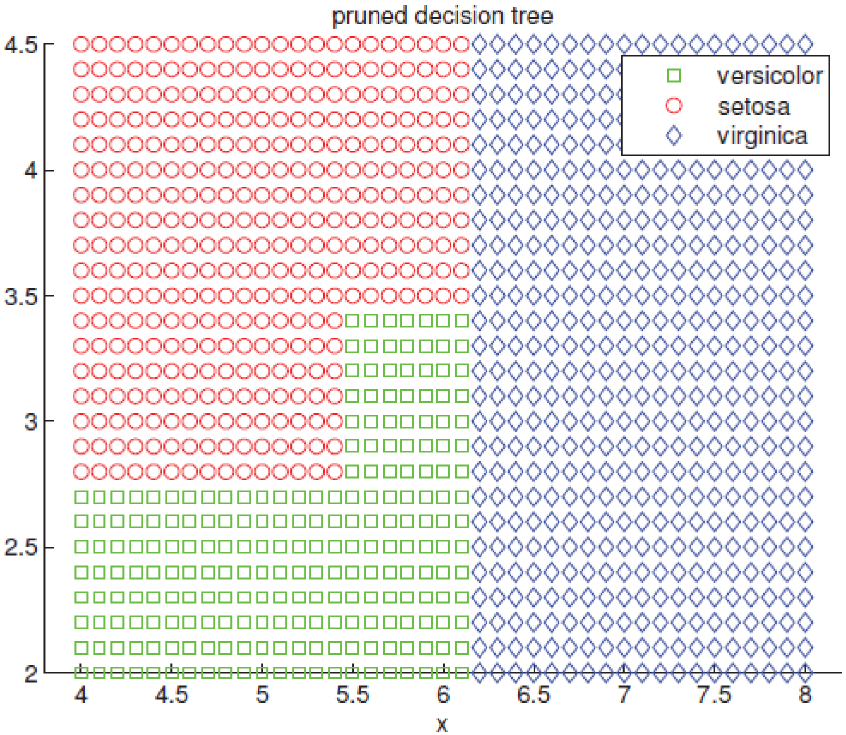
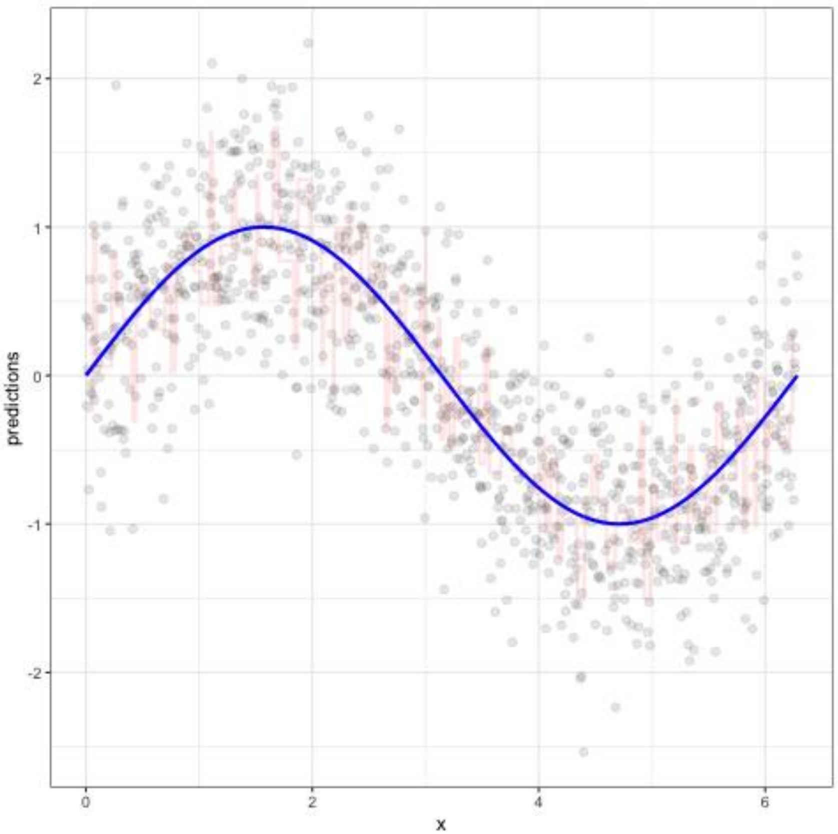


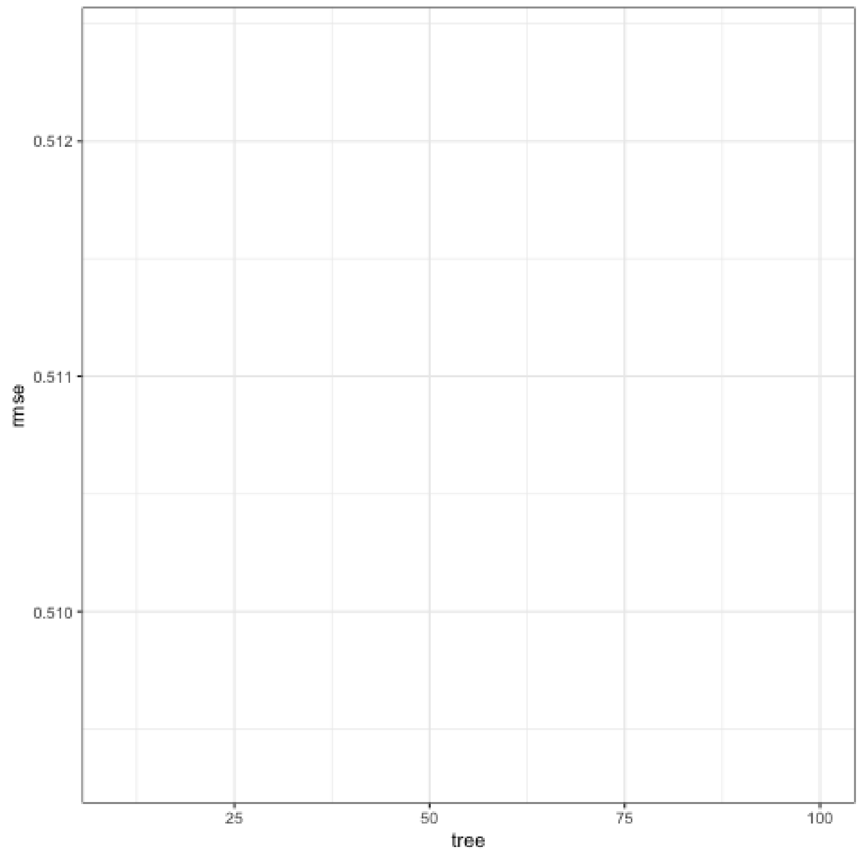
Image source: Kevin Murphy

Rule Post Pruning

1. Convert tree to equivalent set of rules
2. Prune each rule independently of others
3. Sort final rules into desired sequence for use

Ensemble Learning: More Trees





As we add more trees… Prediction error reduces.

Img. Source: https://bradleyboehmke.github.io/

Ensemble Learning: Voting

-Another way to avoid Bias and Overfitting in Decision Trees.

```
from sklearn.ensemble import VotingClassifier
```

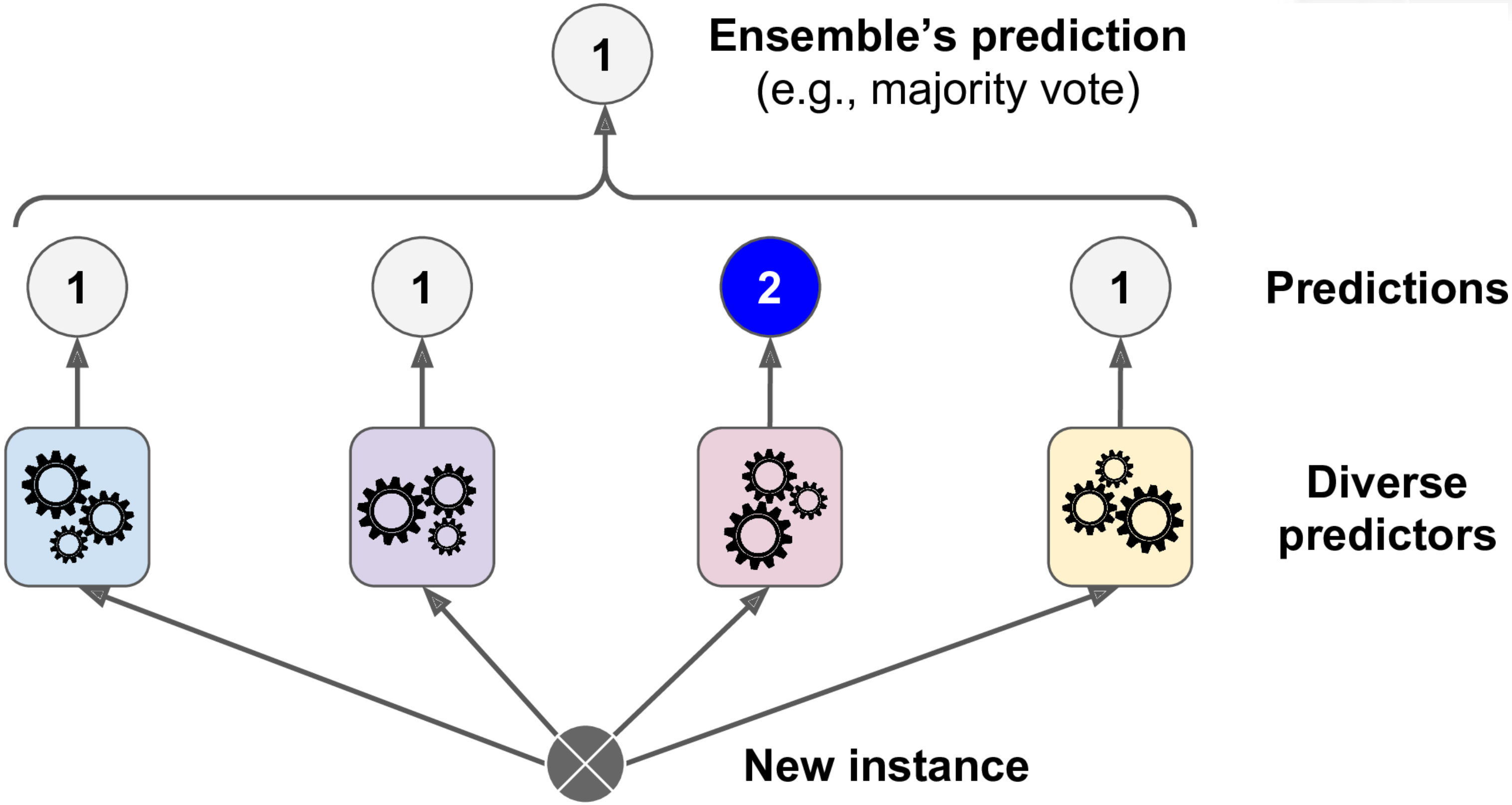


Image source: Aurélien Géron

Recap

- You are allowed to do assignments in groups.
- First ensemble is Voting.

Image source: https://medium.com/

```
from sklearn.ensemble import VotingClassifier
```

Ensemble Learning: Random Forest (RF)

- Use same training algorithm for every predictor in the ensemble, but train them on different random subsets.
- Sampling with replacement: Bagging (Bootstrap Aggregating), and Sampling without replacement: Pasting.

No.	feature1	feature2	feature3	feature4	Class
1	A1	B1	C1	D1	Y1
2	A2	B2	C2	D2	Y2
3	A3	B3	C3	D3	Y3
4	A4	B4	C4	D4	Y4

No.	feat ure1	featu re2	featu re3	feat ure4	Class
1	A1	B1	C1	D1	Y1
1	A1	B1	C1	D1	Y1
3	A3	B3	C3	D3	Y3
3	A3	B3	C3	D3	Y3

No.	feat ure1	featu re2	featu re3	feat ure4	Class
1	A1	B1	C1	D1	Y1
2	A2	B2	C2	D2	Y2
4	A4	B4	C4	D4	Y4
2	A2	B2	C2	D2	Y2

RandomForestClassifier and RandomForestRegressor in Scikit Learn

Random Forest Continued…

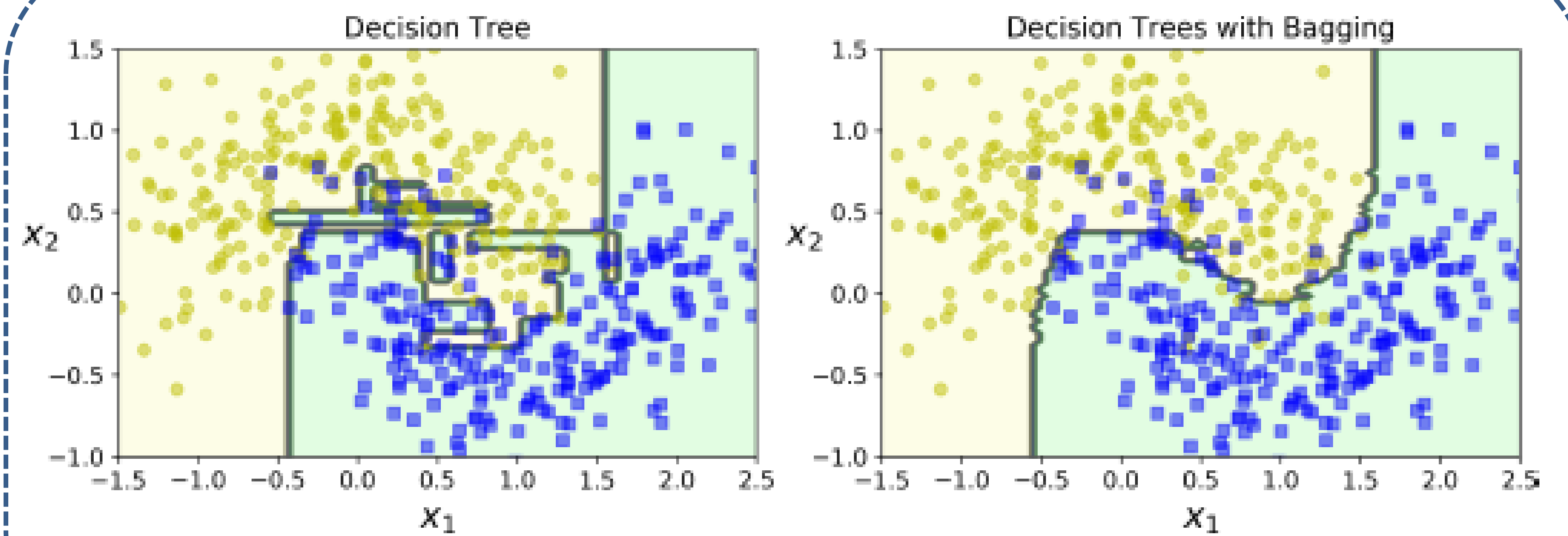


Image source: Aurélien Géron

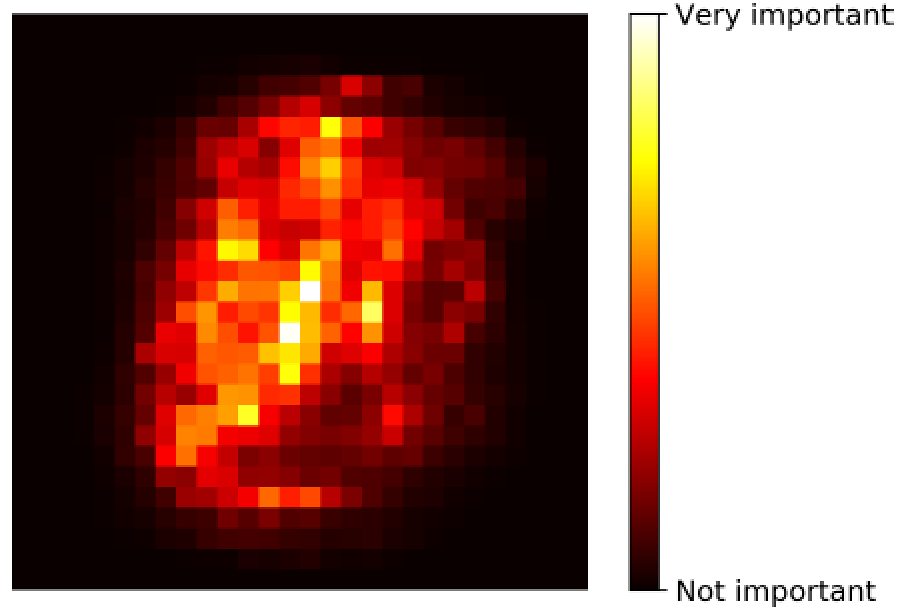
```python
from sklearn.ensemble import RandomForestClassifier

rnd_clf = RandomForestClassifier(n_estimators=500, max_leaf_nodes=16, n_jobs=-1)
rnd_clf.fit(X_train, y_train)

y_pred_rf = rnd_clf.predict(X_test)
```

(Assignment 2)

Random Forest: Feature Importance

- RF can make it easy to measure the relative importance of each feature.
- Scikit-Learn measures a feature's importance by looking at how much the tree nodes that use that feature reduce impurity on an average (across all trees in the forest).

```
>>> from sklearn.datasets import
>>> iris = load_iris()
>>> rnd_clf = RandomForestClassif
>>> rnd_clf.fit(iris["data"], iri
>>> for name, score in zip(iris["
...     print(name, score)
...
sepal length (cm) 0.112492250999
sepal width (cm) 0.0231192882825
petal length (cm) 0.441030464364
petal width (cm) 0.423357996355
```



(MNIST pixel importance)

Image source: Aurélien Géron

(Assignment 2)

Ensemble Learning: Boosting (AdaBoost)

Data Point	Updated Weight
1	0.19
2	0.19
3	0.37
4	0.25

First weak learner

Data Point	Predicted Class
1	0
2	0
3	0
4	1

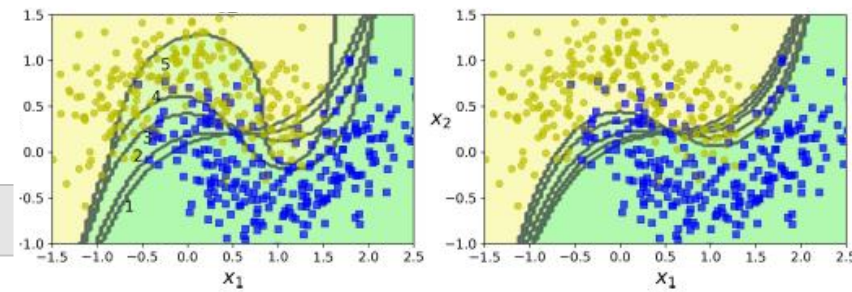
The final prediction is a weighted combination of the predictions of all the weak learners.

$$\text{Final Prediction} = \text{sign}\left(\sum \alpha_i \times \text{Prediction}_i\right)$$

2nd weak learner

Data Point	Predicted Class
1	0
2	0
3	1
4	1

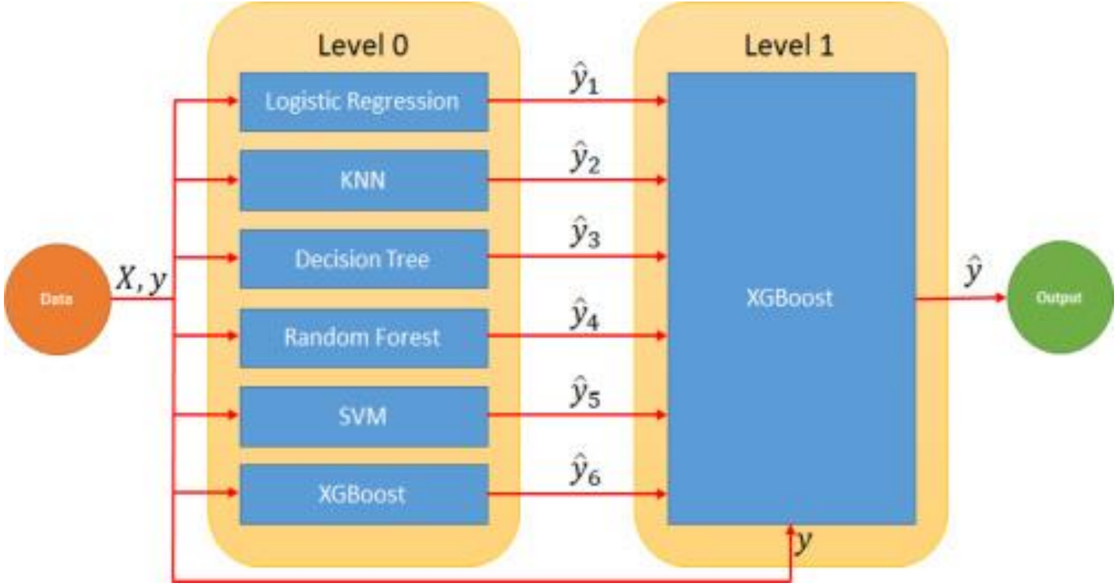
(Decision boundaries of consecutive predictors)



Data Point	Feature 1	Feature 2	Class (Label)
1	1	2	0
2	2	3	0
3	3	4	1
4	4	5	1

Gradient Bosting: Instead of tweaking the instance weights at every iteration like AdaBoost, it tries to fit the new predictor to *residual errors (actual-predicted)* made by the previous predictor. What is XGBoost?

Ensemble Learning: Stacking



(Image source: Alireza Ghasemeih et al.)

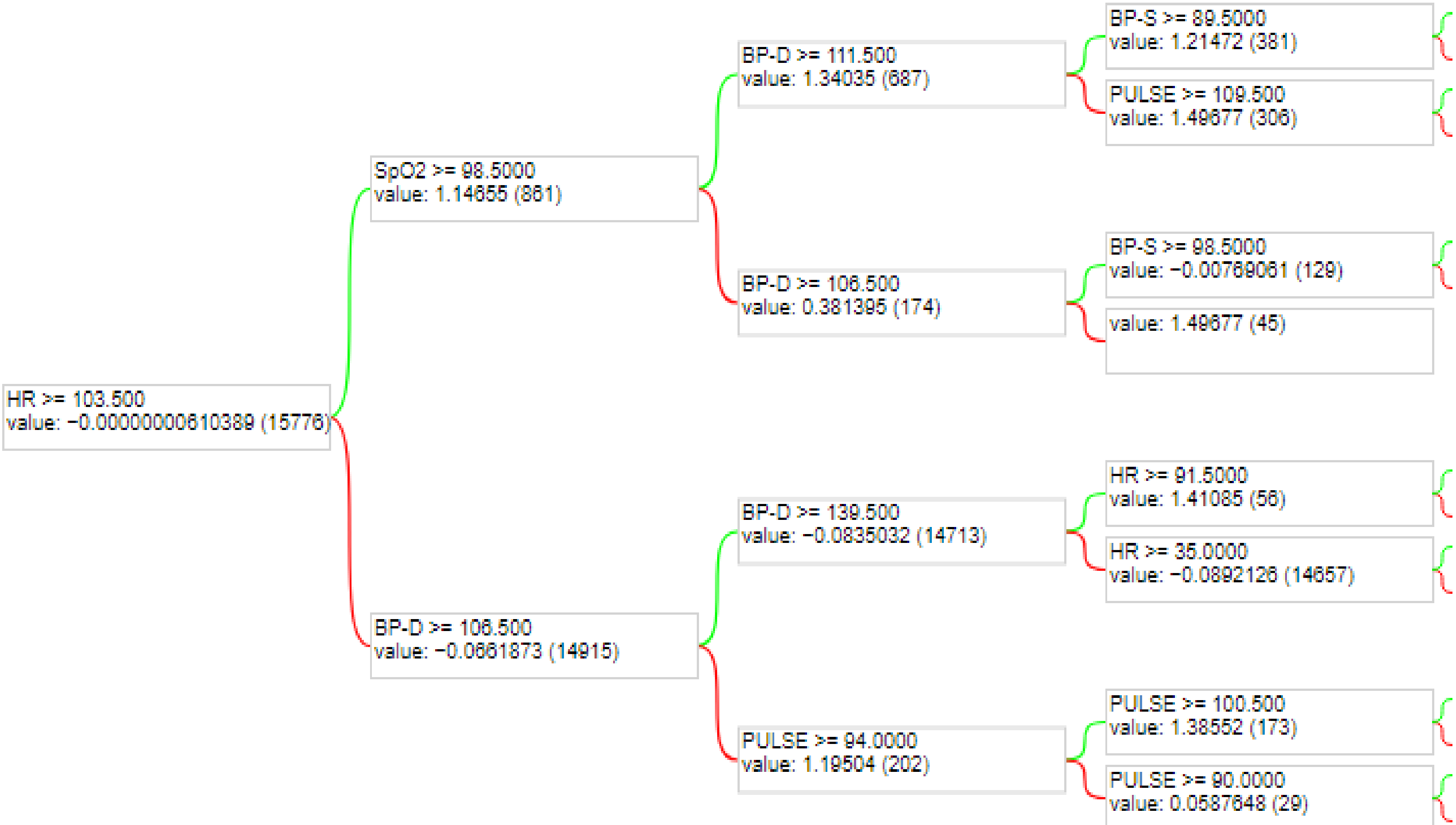
TensorFlow's Decision Forest (Assignment 2)

	RESP	BP-S	BP-D	SpO2	HR	PULSE	Anomaly
4913	21	87	129	100	78	78	0
9338	22	87	130	100	83	82	1
24210	23	75	108	99	87	86	0
18790	27	83	119	98	91	91	0
16066	19	81	115	97	95	95	0

(MIMIC Dataset: Medical Information Mart for Intensive Care from MIT's Computational Physiology Lab.)

https://ieee-dataport.org/documents/ble-wban-rf-real-world-dataset-ble-devices-human-centric-healthcare-environments

TensorFlow's Decision Forest (Assignment 2)



Submission deadline: 11.09.2024

Thank You!