ε.δε.μ²
ΕΠΙΣΤΗΜΗ ΔΕΔΟΜΕΝΩΝ & ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Decision Trees

1. *C*lassification *A*nd *R*egression *T*rees (CART), Gini Index

2. Random Forests

3. Bagging (*B*ootstrap & *agg*regating) Algorithms

Prof. Vasilis Maglaris

maglaris@netmode.ntua.gr

www.netmode.ntua.gr

Room 002, New ECE Building

Tuesday May 20, 2025

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Decision Trees for Non-Parametric Supervised Learning

Overview of Decision Tree Supervised Learning

https://dataaspirant.com/how-decision-tree-algorithm-works/

Decision Trees are ***predictive models*** that respond to observed ***input*** examples with ***output target*** conclusions by placing them in ***predictive regions***, defined by ***recursively binary splitting***

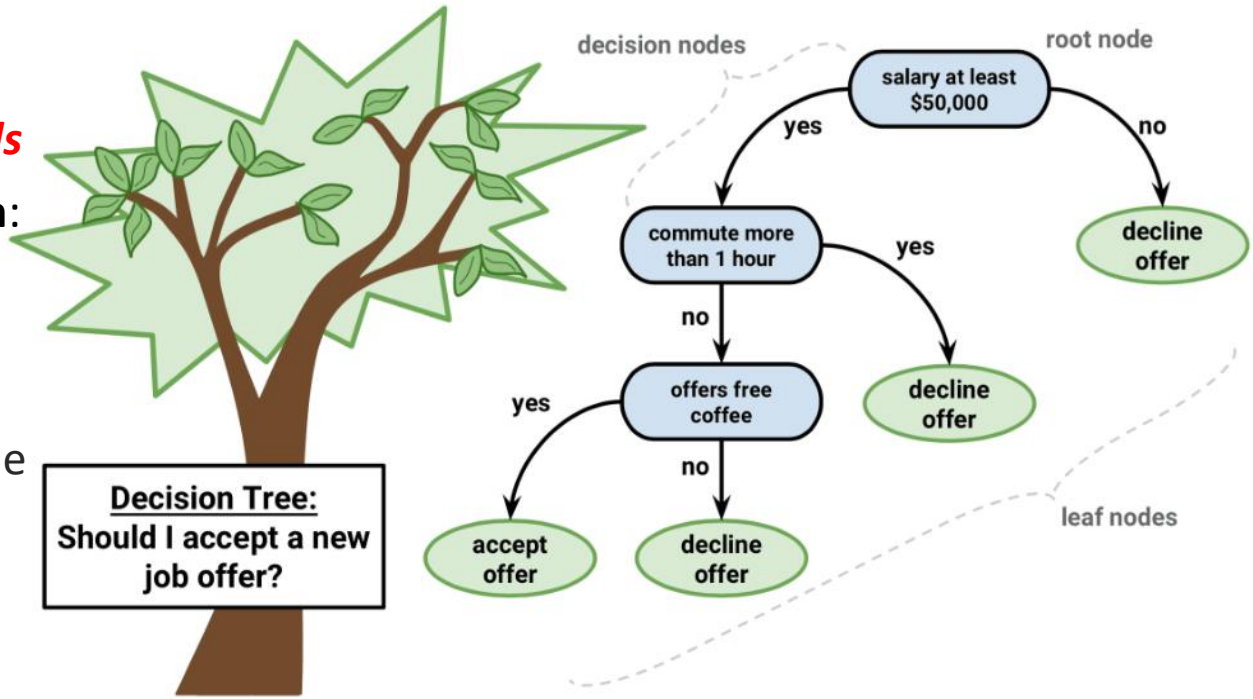
- **Regression Trees** respond with an output ***value*** of an observed input ***predictor*** by sequentially placing it in ***predictive regions*** depending on the ***numerical range*** of an ***attribute*** (feature)
- **Classification Trees** respond with the most probable ***categorical class*** for an input ***predictor*** that represents a ***predictive region*** of an input ***attribute***

Tree Configuration:

Internal nodes correspond to ***attributes***, ***leaf*** nodes to ***labels***

Supervised Learning Algorithm:

1. Place a key attribute of the training dataset at the ***root***
2. Split the set into ***two***. Each subset contains data with the same value for an attribute
3. Repeat steps 1 & 2 on each subset until you find ***leaf nodes*** in all tree branches



Decision Tree classifier, **Image credit:** www.packtpub.com

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Non-Parametric Supervised Learning for Decision Trees (1/3)

J. Gareth, D. Witten, T. Hastie, R. Tibshirani 2013, *"An Introduction to Statistical Learning"*
https://hastie.su.domains/ISLR2/ISLRv2_website.pdf (Ch. 8)

- A **Binary Decision Tree** receives input vectors $\mathbf{x} = [x_1\ x_2\ \dots x_m]^{\mathrm{T}}$ with x_i a metric (*predictor value*) of its attributes (*features*) and decides on placing them to *internal* and finally *leaf* nodes by sequential comparisons. Nodes in the tree implement binary decisions depending on specific *predictor values*

- The **Decision Tree** is pre-configured by considering *training examples* (patterns) with *categorical* or *numerical values* (*predictor values*) of their *attributes,* that result to *output target values* $\mathrm{y}(i)$, estimated by *labels* $d(i)$. During the *Training Phase* of *supervised learning*, for the N training vectors in $\boldsymbol{\mathcal{D}} = \{(\mathbf{x}(1), d(1)), \dots, (\mathbf{x}(N), d(N))\}$ the tree is traversed from the *root* to a *leaf node* that signifies *target* prediction $\mathbf{x}(i) \rightarrow y(i) \cong d(i)$

- **Traversal paths** are minimized based on prior statistical knowledge of the sample space

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Non-Parametric Supervised Learning for Decision Trees (2/3)

J. Gareth, D. Witten, T. Hastie, R. Tibshirani 2013, *"An Introduction to Statistical Learning"*
https://hastie.su.domains/ISLR2/ISLRv2_website.pdf (Ch. 8)

- There are two kinds of *Decision Trees*:

  - **Regression Trees** group *continuous numerical attributes* of examples into M non-overlapping *prediction regions* $R_1, R_2, \dots, R_M$. Trees are structured based on supervised learning, with continuous *labels* $d(i) \cong y(i)$. The tree learning algorithm sequentially selects which node (*attribute*) will be subject to further splitting in **two** parts based on *threshold* values. This may require *exhaustive* search or may be approximated by *pruning* of unlikely options. The **Optimal Predictive Values** $\mathcal{C}_k$ of a region R_k will be the *average label* of its examples

  - **Classification Trees** lead to *leaf nodes* that determine for each *region* a pre-defined *categorical class* $\mathcal{C}_K$: $\mathbf{x}(i) \rightarrow y(i) \Rightarrow \mathcal{C}_K$. *Binary* $\mathrm{y}(i) \in \{0,1\}$ denotes classification of $\mathbf{x}(i)$ into two classes, e.g. $\mathcal{C}_1 = \boldsymbol{yes}, \mathcal{C}_2 = \boldsymbol{no}$

- New *test* input elements are predicted to *region* or *class* membership by traversing the *binary decision tree* from the *root* to a *leaf node* that corresponds to a *final prediction*

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Non-Parametric Supervised Learning for Decision Trees (3/3)

J. Gareth, D. Witten, T. Hastie, R. Tibshirani 2013, *"An Introduction to Statistical Learning"*
https://hastie.su.domains/ISLR2/ISLRv2_website.pdf (Ch. 8)

- A **leaf** in the tree denotes that its **upstream nodes** considered all relevant attributes, thus a final prediction is reached on the *class* (*classification* tree) or *range of numerical* attributes (*regression* tree) of $\mathbf{x}(i)$ by following decisions in the downstream path

- **The downstream paths** will hopefully avoid decisions on irrelevant attributes and will contain (*statistically*) as few as possible *internal* nodes as dictated by a good training algorithm (**CAUTION**: Overfitting Risk)

- **Popular Use Cases**: Pattern recognition, processing - classification - search of text and images, environmental studies, biomedical analyses, medical diagnoses, search engines, cyber-attack detection and mitigation (e.g. DNS attacks, antivirus, anti-spamming…)

- **Advantages**: Easy *non-parametric* implementation, intuitive *reasoning* & *explainability*, no (or limited) hyper-parameters to be tuned…

- **Disadvantages**: Need for stable *labeled training datasets*, accuracy problems, sensitive to small variations of input characteristics, risk of *overfitting*…

Ovefitting can be mitigated by combining several decision trees (**Random Forests**)

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Example of Recursive Partitioning for Regression Tree

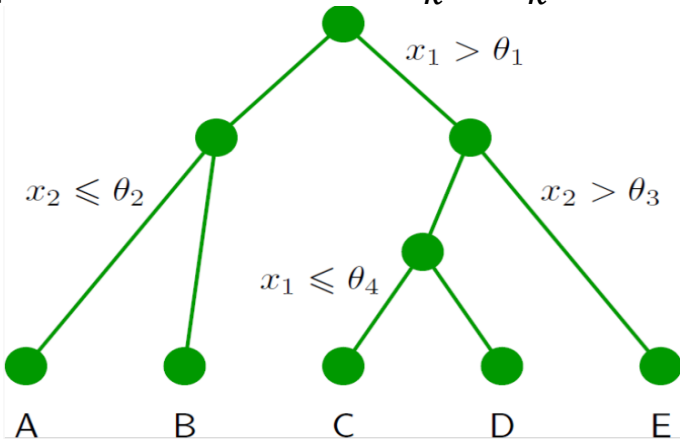
Christopher M. Bishop 2009, *"Pattern Recognition and Machine Learning"*
https://dl.acm.org/doi/10.5555/1162264 (Ch. 14.4)

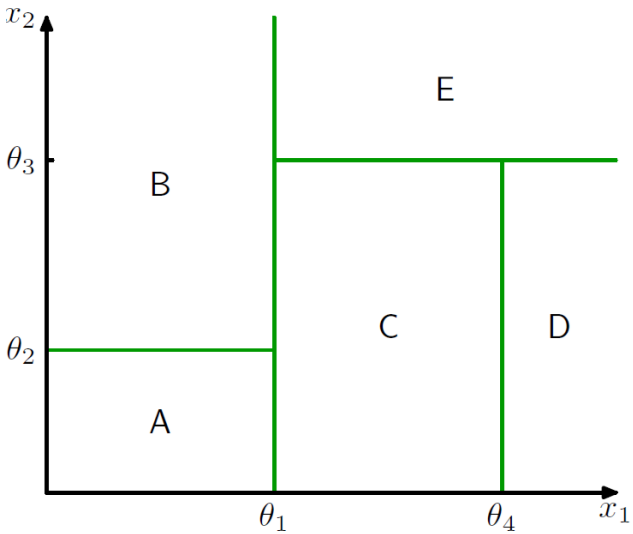
Training of Binary Regression Tree

- Assume m - dimensional *input* examples $\mathbf{x} = [x_1\ x_2\ \dots x_m]^{\mathrm{T}}$ and place them into M *regions* $R_k,\ \ k = 1,2,\dots,M$ by recursively splitting regions in two, across axis-aligned boundaries specified by *Threshold Parameters* θ_k
- Let a *Training Sample* of N vectors $\mathbf{x}_j \to y_j$ with *continuous* labels $d_j \cong y_j,\ j = 1,2,\dots,N$
- Structure of the Tree:
  - Set the *root* to correspond to the whole input space
  - Select *split variables* & *thresholds* to recursively place a *training example* $\mathbf{x}_j$ in *region* R_K
  - The *optimal predictive variables* $\mathcal{C}_K$ are the *average of labels* d_j of all N_k vectors $\mathbf{x}_j \in R_k$:

$$\mathcal{C}_k = \frac{1}{N_k} \sum_{\mathbf{x}_j \in R_k} d_j$$

- The optimal prediction of all $\mathbf{x} \in R_k$ is $\mathcal{C}_k$





STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Configuration of Classification Trees with Binary Target Value $y(i) \in \{0,1\}$

- A node m of the tree corresponds to a subtree for *region* $\boldsymbol{R}_m$ as defined by the *range* of *predictor variables* of input vector $\mathbf{x}(i) \in \boldsymbol{R}_m$. The *analogy* (relative frequency) of *target values* $y(i)$ for input vectors $\mathbf{x}(i) \rightarrow (y(i) = k)$ in the N_m elements in *region* $\boldsymbol{R}_m$ is:

$$\hat{p}_{mk} \triangleq \frac{1}{N_m} \sum_{\mathbf{x}(i) \in \boldsymbol{R}_m} I(y\,(i) = k)$$

- The *majority* of classifications for $\mathbf{x}(i) \in \boldsymbol{R}_m$ occurs for a *class* given by

$$\underset{k}{\operatorname{argmax}}\, \hat{p}_{mk}$$

- Every different class assignment is considered as an *impurity*. A metric for sub-tree configuration (*attribute selection*) is the **Gini Index**

$$1 - \sum_{k} \hat{p}_{mk}^2$$

- The input *attribute* that will determine a subsequent split to subtrees result from selecting the minimum of the **Gini Index** (GI) for the residual attributes (features)

- For 2 classes $(\mathcal{C}_1, \mathcal{C}_2)$, $GI(p) = 2p(1-p) = 1 - (1-p)^2 - p^2$ where p is the analogy of *classification errors* (*impurities*) of $\{\mathbf{x}(i), d(i) = 0\}$ in sub-tree m to *target* $y(i) = 1 \neq d(i)$. In case of **absolute consistence** (no errors) $p = 0$ or $p = 1$ and $GI(p) = 0$ (**minimum value**). The **highest inconsistence** occurs for $p = 0.5$ yielding $GI(p) = 0.5$ (**maximum value**)

Note Another widely used to select splitting attributes is the *maximization* of the *Information Gain* of the *Entropy Index* $-p\log_2(p) - (1-p)\log_2(1-p)$ of sub-tree alternatives

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Example of Classification Tree Configuration Using the Gini Index (1/2)

https://dataaspirant.com/how-decision-tree-algorithm-works/

The *training sample* consists of $N = 16$ input vectors (*records*), each with 4 **continuous numerical attributes** (*feature values*, *predictors*) A, B, C, D and **categorical** dual output (*label*, *target*):

$$\mathrm{E} \in \{\text{positive}, \text{negative}\}$$

The *predictor* values of input *Attributes* are **arbitrarily** *classified* in 2 **categories** (*ranges*) depending on their **price**:

A	B	C	D
≥ 5.0	≥ 3.0	≥ 4.2	≥ 1.4
< 5.0	< 3.0	< 4.2	< 1.4

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Gini Index GI **for Attribute** A:

$\mathrm{A} \geq 5.0: \ 12/16, \mathrm{A} < 5.0: \ 4/16$

$\mathrm{A} \geq 5.0$ & E positive: $5/12$

$\mathrm{A} \geq 5.0$ & E negative: $7/12$

$GI(5,7) = 1 - (5/12)^2 - (7/12)^2 = 0.486$

$\mathrm{A} < 5.0$ & E positive: $3/4$

$\mathrm{A} < 5.0$ & E negative: $1/4$

$GI(3,1) = 1 - (3/4)^2 - (1/4)^2 = 0.375$

$GI(\mathrm{A}) = 12/16 \times 0.486 + 4/16 \times 0.375 = \mathbf{0.45825}$

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Example of Classification Tree Configuration Using the Gini Index (2/2)

https://dataaspirant.com/how-decision-tree-algorithm-works/

Gini Index GI for Attribute B:

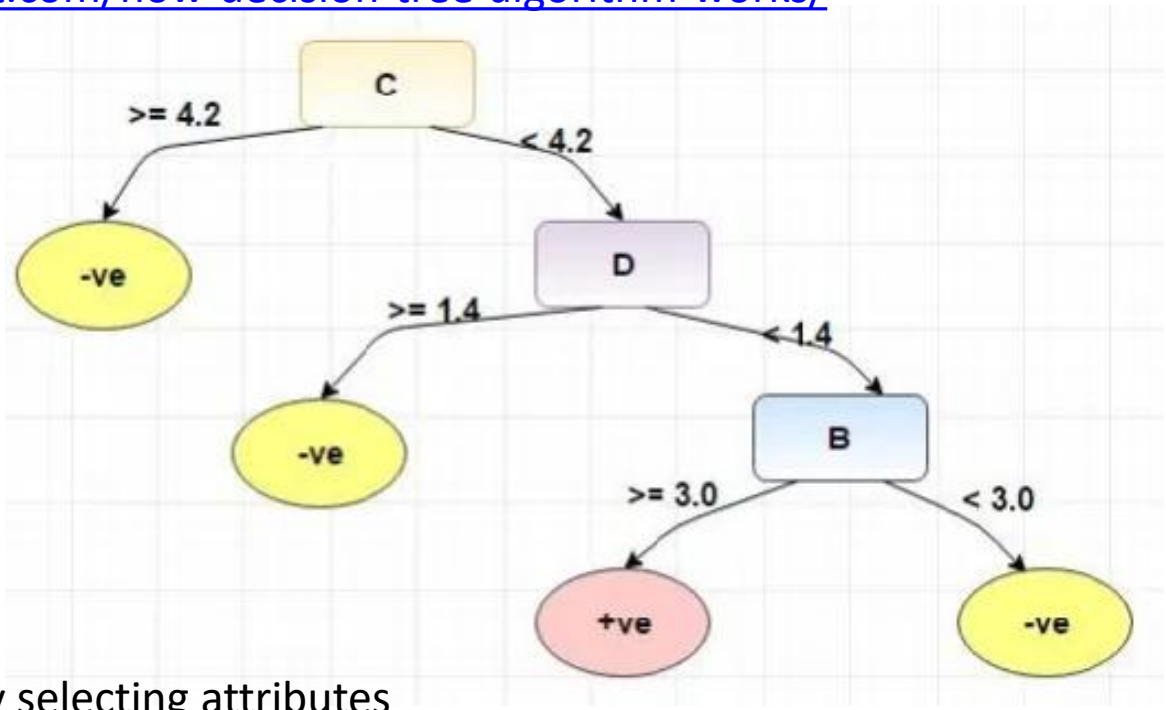
Similarly, as for **Attribute** A we have $GI(\mathrm{B}) = \mathbf{0.3345}$

Gini Index GI for Attribute C:

Similarly, as for **Attribute** A we have $GI(\mathrm{C}) = \mathbf{0.2}$

Gini Index GI for Attribute D

Similarly, as for **Attribute** A we have $GI(\mathrm{D}) = \mathbf{0.273}$



- The **Decision Tree** is configured by selecting attributes in opposite order of the GI values: C, D, B
- Attribute A with $GI(\mathrm{A}) = \mathbf{0.45825}$ does not affect the tree configuration process

Attribute	A	B	C	D
Lower Value	≥ 5.0	≥ 3.0	≥ 4.2	≥ 1.4
Upper Value	< 5.0	< 3.0	< 4.2	< 1.4
Gini Index	$\mathbf{0.45825}$	$\mathbf{0.3345}$	$\mathbf{0.2}$	$\mathbf{0.273}$

Classification Tree Example: Hepatitis Prediction (1/6)

Based on Presentaation of ***Adele Cutler*** (Utah State University), " **Random Forests for Regression and Classification**", Ovronnaz, Switzerland, Sep. 2010 https://math.usu.edu/adele/randomforests/ovronnaz.pdf

A Classification Tree

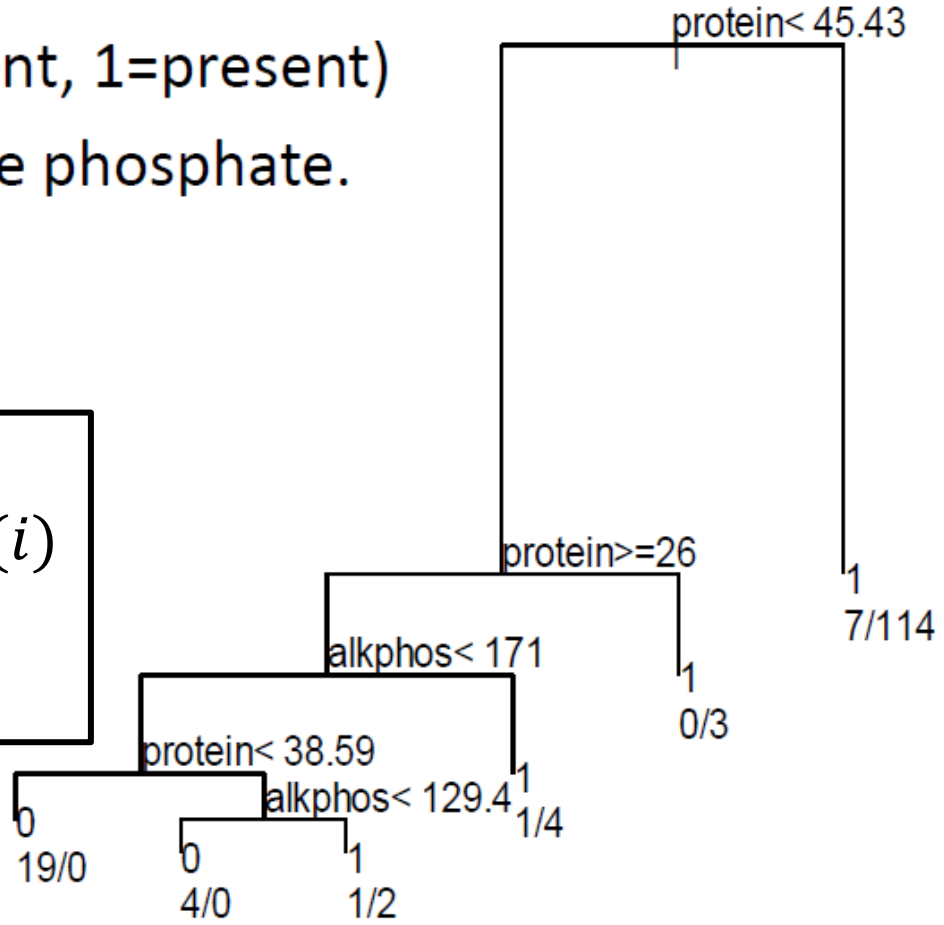
Predict hepatitis (0=absent, 1=present) using protein and alkaline phosphate.

"Yes" goes left.

Training Sample

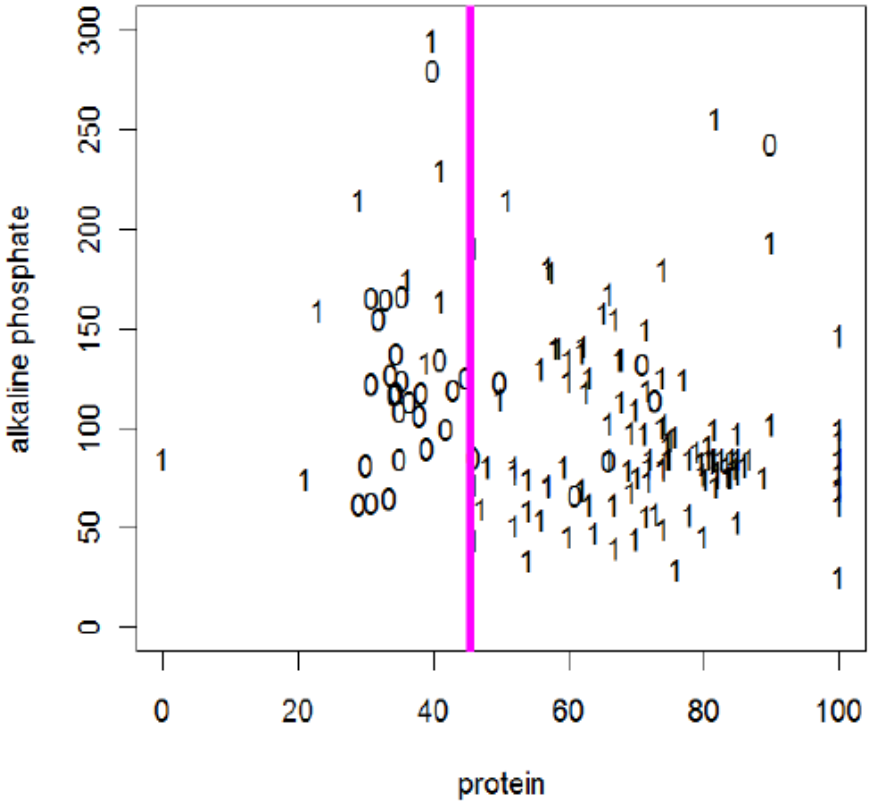
155 Labeled Sample Elements: $\mathbf{x}(i)$

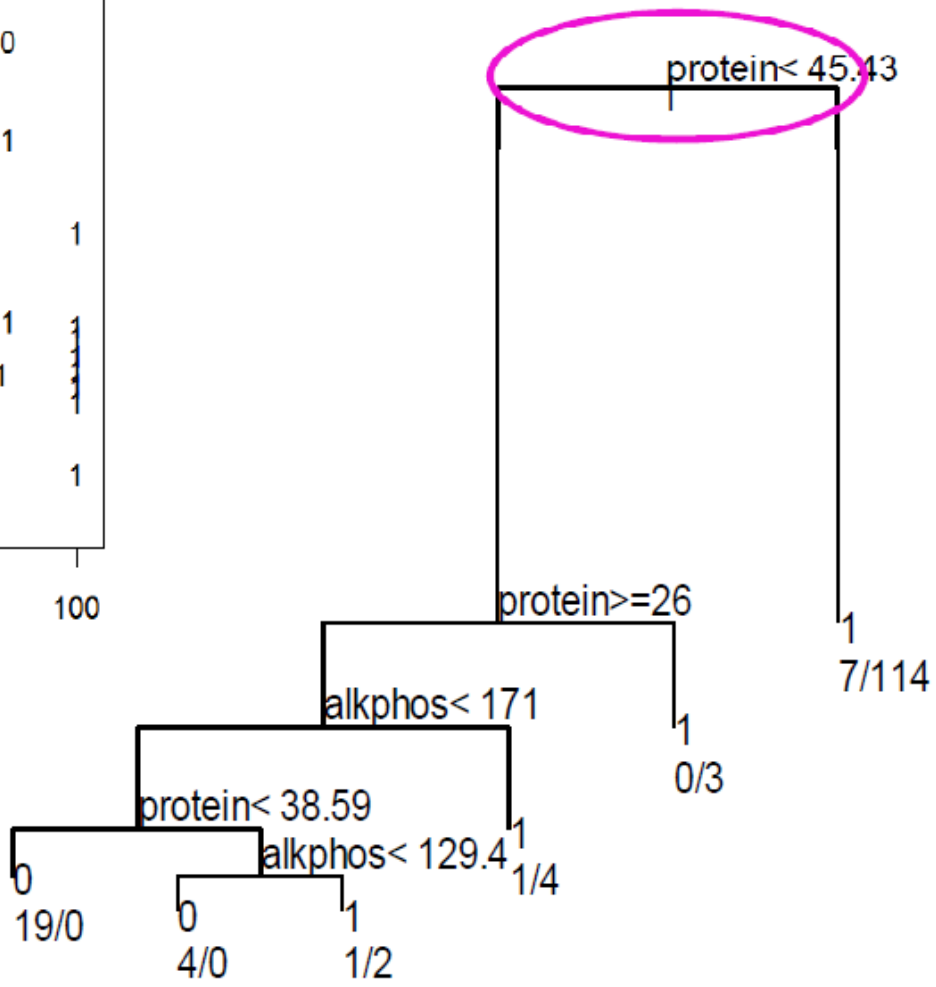
- 123 present: $y(i) = 1$
- 32 absent: $y(i) = 0$



STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

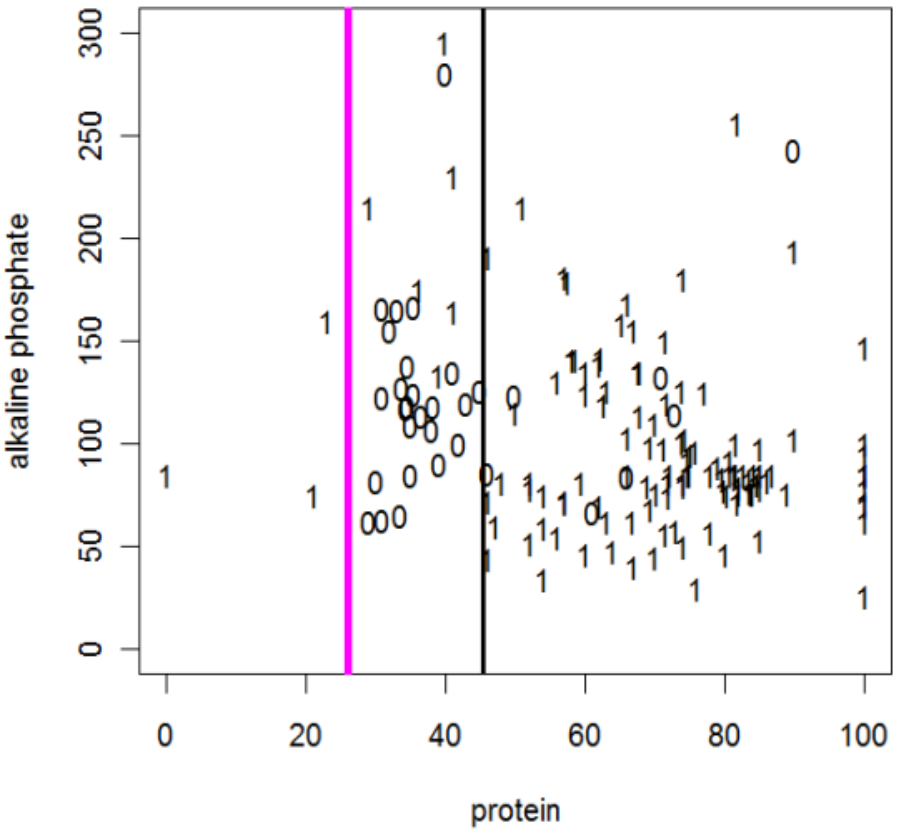
Classification Tree Example: Hepatitis Prediction (2/6)

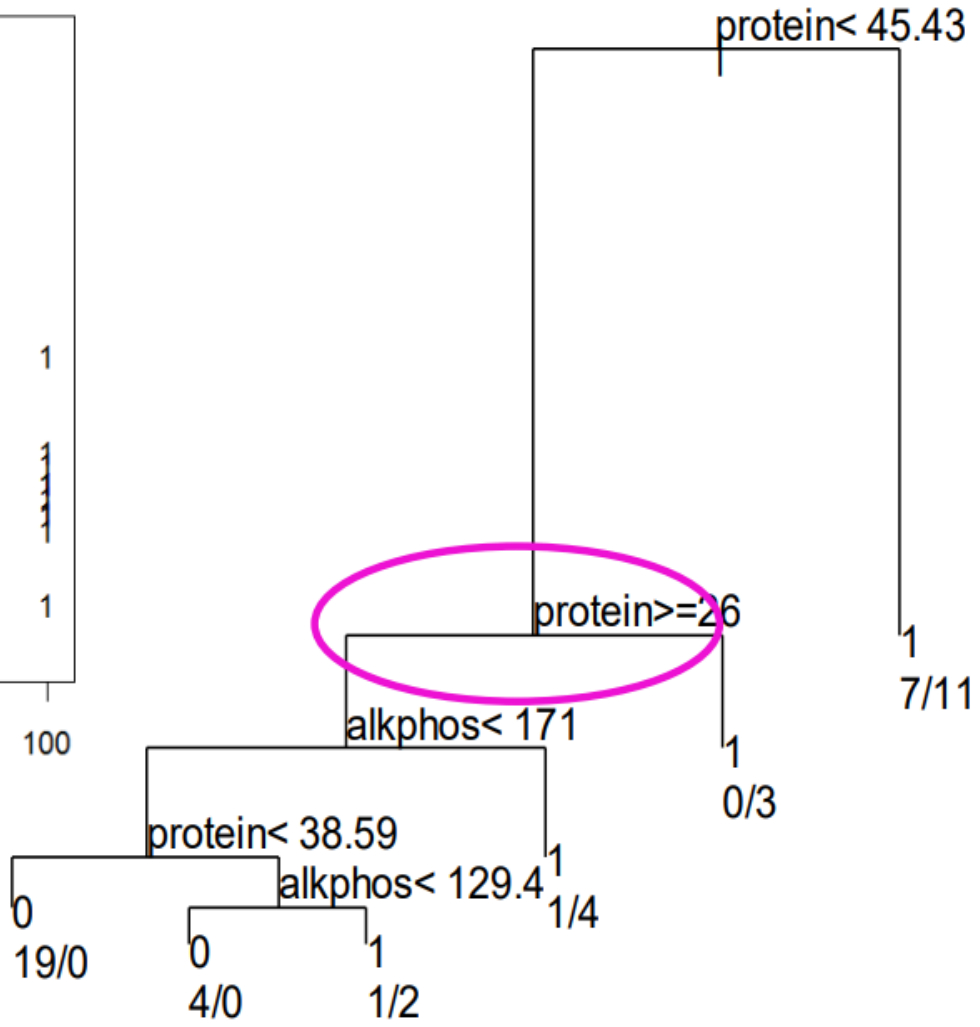




STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

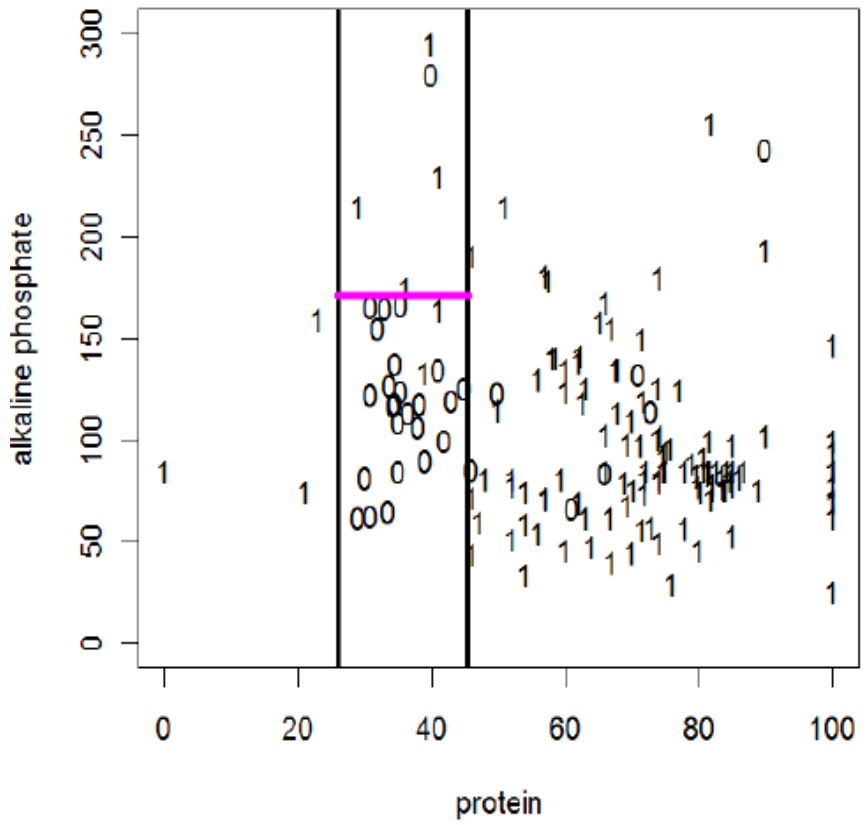
Classification Tree Example: Hepatitis Prediction (3/6)

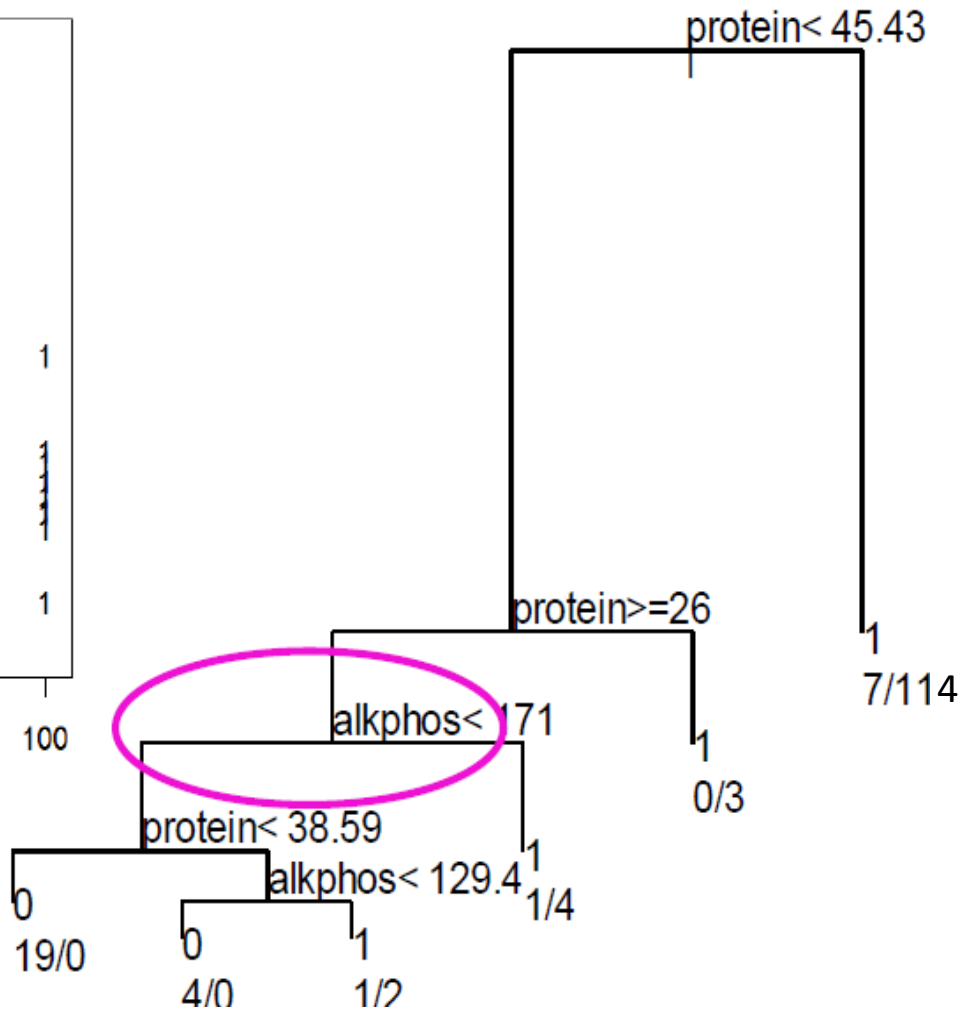




STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

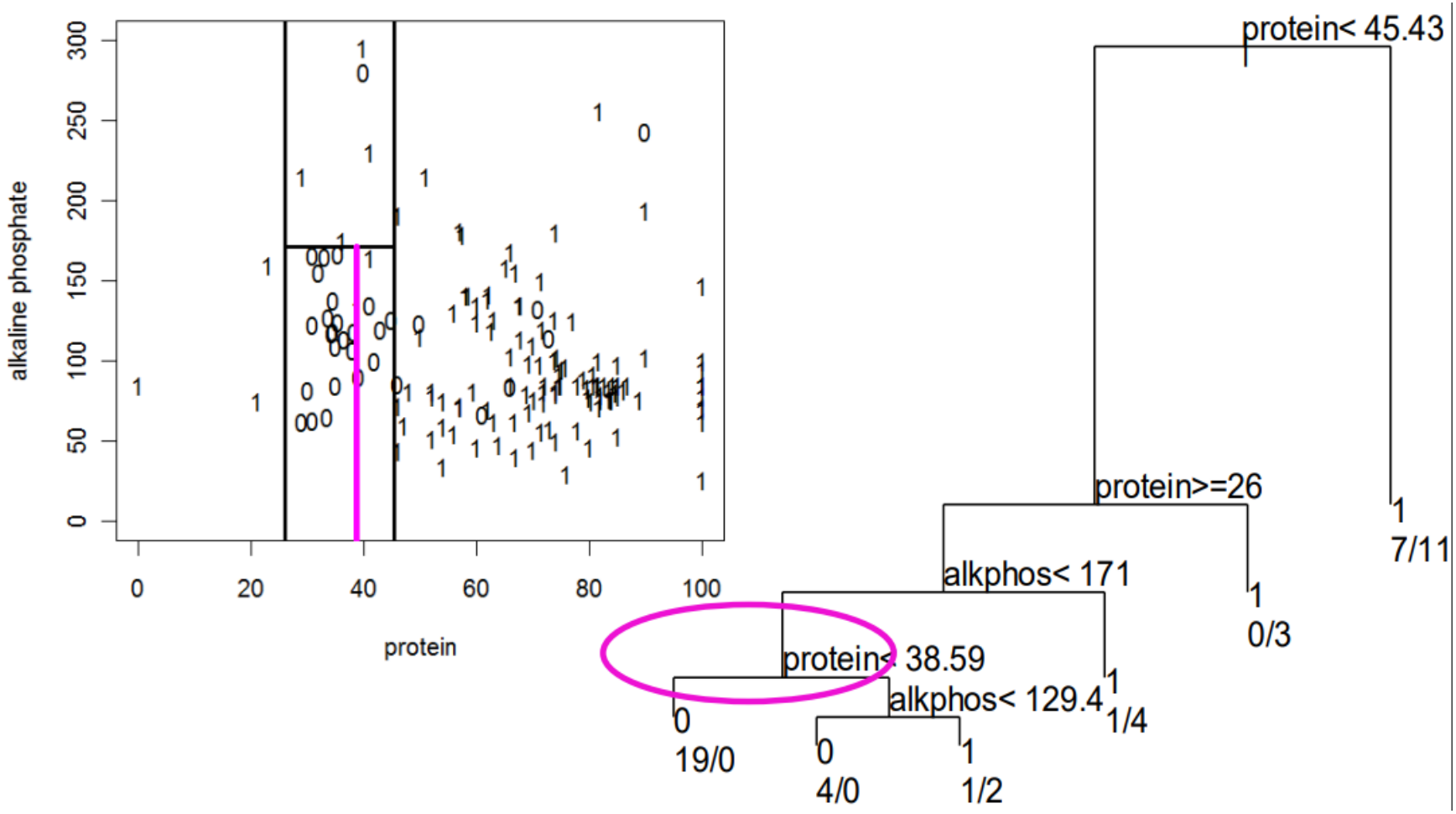
Classification Tree Example: Hepatitis Prediction (4/6)





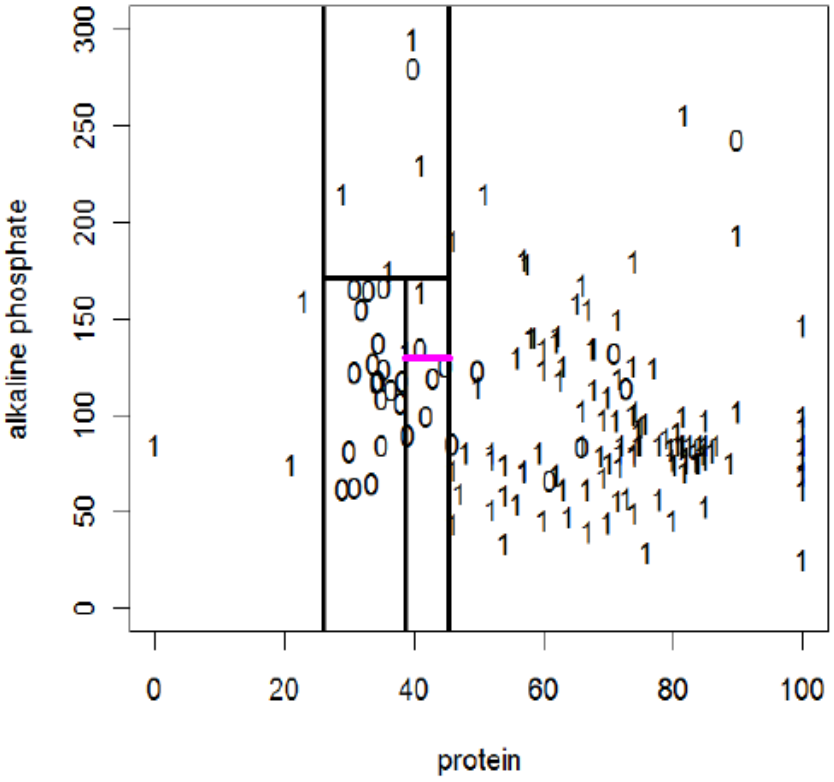
STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

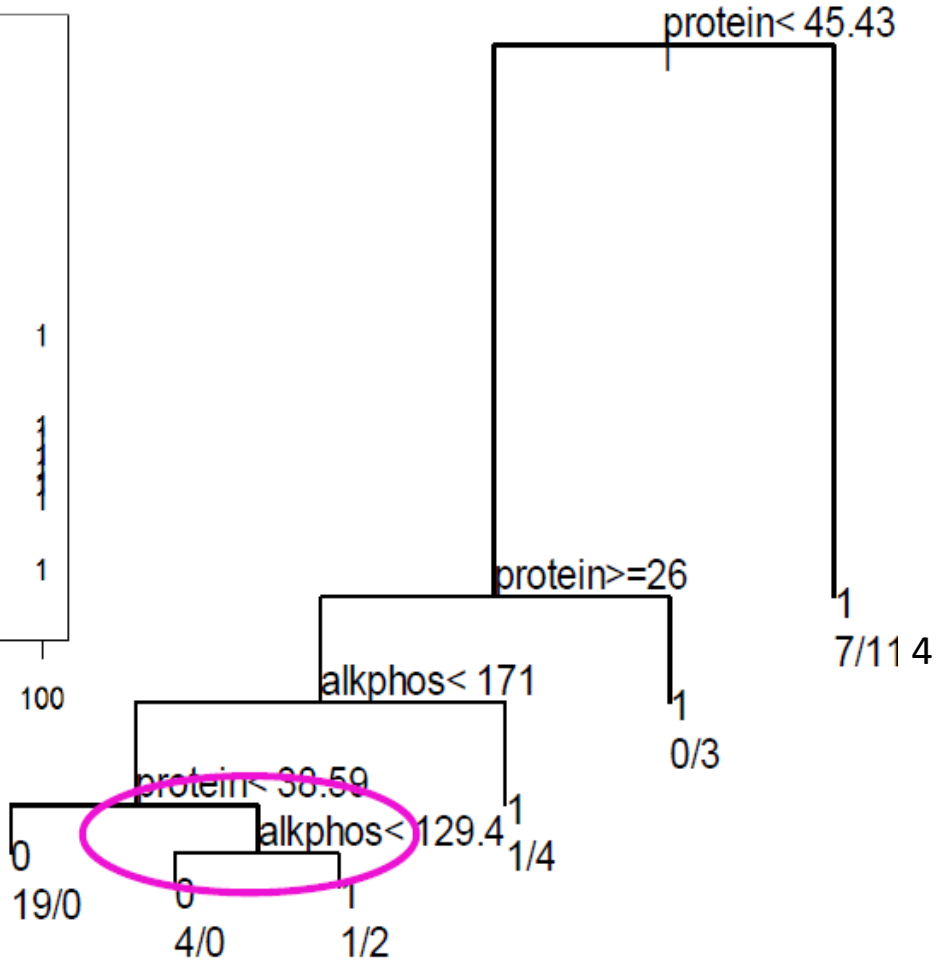
Classification Tree Example: Hepatitis Prediction (5/6)



STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Classification Tree Example: Hepatitis Prediction (6/6)





The *predictor* $protein$ uniquely defines splitting across its large or small values
The *predictor* $alkaline\ phosphate$ is considered only for middle values of $protein$

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Configuration of Random Forests - Bootstrap Aggregating (Bagging)

Leo Breiman, "**Random Forests**", Machine Learning, 45, 5-32, Kluwer Academic Publishers 2001
https://link.springer.com/content/pdf/10.1023/A:1010933404324.pdf
Leo Breiman, "**Bagging Predictors**", Machine Learning, 24, 123-140, Kluwer Academic Publishers 1996
https://link.springer.com/content/pdf/10.1007/BF00058655.pdf

- A *Random Forest* consists of R *decision trees* that *independently* predict a target $\mathrm{y}(i)$. The final decision (*prediction*) is reached by **aggregation** of *target values* $\mathrm{y}(i)$: *Voting* for *classification*, *averaging* for *regression*
- The supervised learning algorithm follows the **Bootstrap Aggregating** (***Bagging***) of *Leo Breiman* (1996)
- We randomly select R *bootstrap samples*, subsets of a *labeled* training sample $\boldsymbol{\mathcal{D}} = \{(\mathbf{x}(1), d(1)), \ldots, (\mathbf{x}(N), d(N))\}$. Default choice is $R = 500$ but results are usually satisfactory for small values (e.g. $R = 20$)
- Selection of *examples* (*observation*) into subsets is accomplished with independent random calls and with **replacement of selected observations** (*sampling with replacement*). Thus, an *observation* in a *bootstrap sample* can be selected more than once. *Examples* that are not selected in any of the R subsets are considered as *out of bag* (**oob**) *observations* (too many **oob**'s may lead to unfeasible predictions)
- The *bootstrap trees* are configured in all depth via the **CART** (*Classification And Regression Trees*) algorithm but with a *reduced* number of features (*predictor attributes*), pruned randomly and independently in each node of the trees

STOCHASTIC PROCESSES & OPTIMIZATION IN MACHINE LEARNING

Validation and Performance of Random Forests & Bagging Algorithm

- Evaluation via **cross-validation**: Any large *labeled* sample can be divided into a significant number of subsets $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \ldots$ A single $\mathcal{D}_k$ may be set aside as a **test set** and the *labels* of its elements can be used to benchmark predictions based on the other subsets, used as *supervised training sets*

- Performance of *Random Forests* (**RF**) can be assessed by counting errors of **RF Predictors** in *out of bag* (**oob**) *observations*

- Many benchmarks demonstrated that **RF**'s do not inherit typical problems of *Decision Trees*: They exhibit decent prediction accuracy with small *variance*, are not sensitive in noisy measurements of *predictor variables*, do not cause **overfitting**

- In general, they are considered a promising **interpretable** model for **non-parametric** classification, prediction and comparative evaluation of *attributes* of multi-dimensional samples. Nevertheless, they rely on the availability of reliable *labeled* training samples to enable **supervised learning**

- In case of very large models, the memory and processing needs of **RF**'s may require deployment of high-performance H/W (**GPU**) and/or *cloud services*