LECTURER: TAI LE QUY

MACHINE LEARNING

SUPERVISED LEARNING

UNIT 5

DECISION & REGRESSION TREES

- Explain the concept of **decision** and **regression trees**.
- Explain the power of the **ensemble methods** widely used in practice.
- Define **bagging** and **boosting**.
- Apply two very popular ensemble models on your own with the use of **Python**.

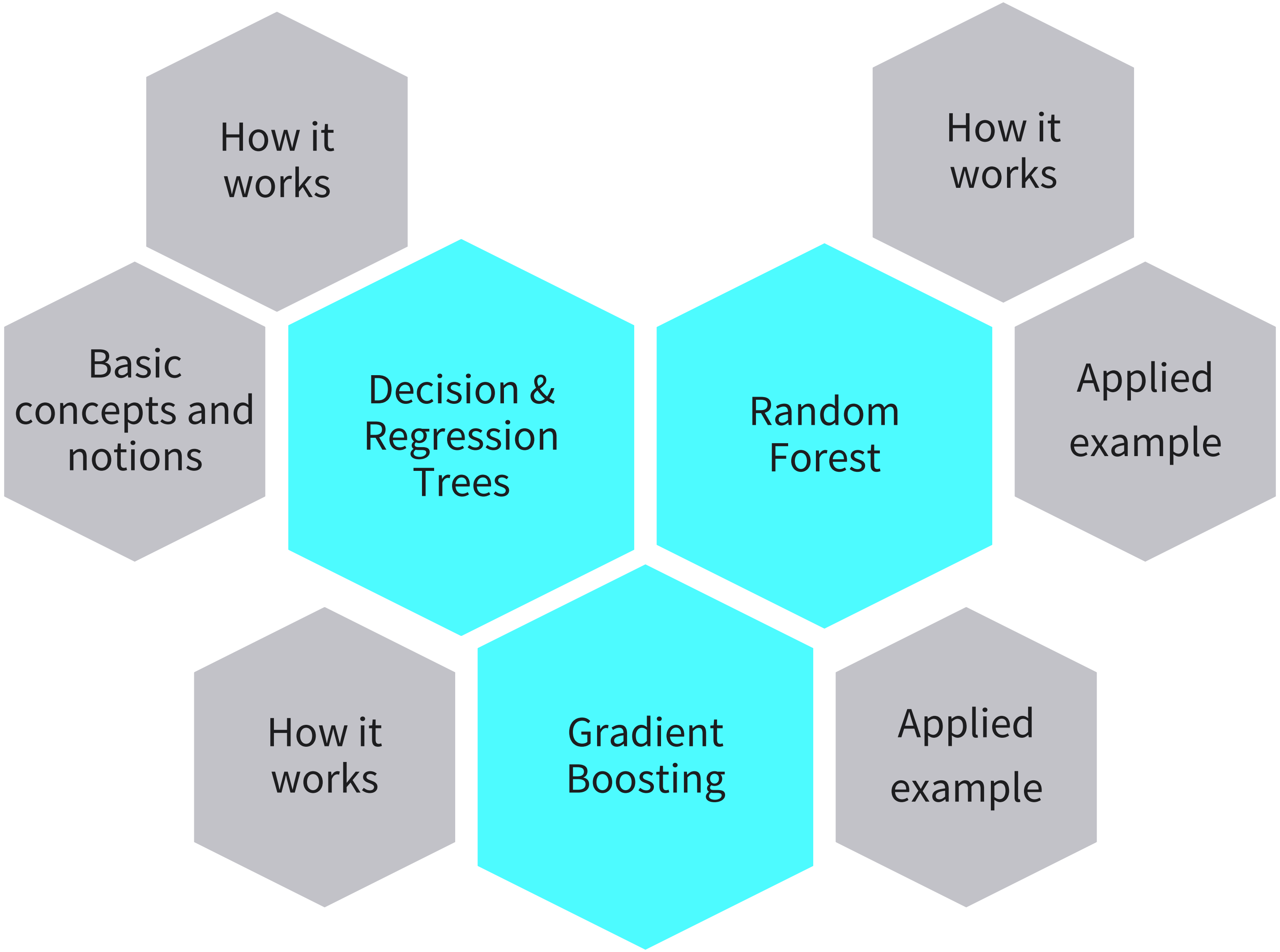
1. What are **split criteria** used for?

2. What are **stop criteria** used for?

3. What is meant by the term "**ensemble model**"?

UNIT CONTENT

Img. 1: Decision & regression trees



Source of image 1: Christian Müller-Kett, 2021

- One of the most popular classification methods
- DTs are included in many commercial systems nowadays
- Easy to interpret, human readable, intuitive
- Simple and fast methods.
- Many DT induction algorithms have been proposed
  - ID3 (Quinlan 1986)
  - C4.5 (Quinlan 1993)
  - CART (Breimanet al 1984)

Source of image: Course book

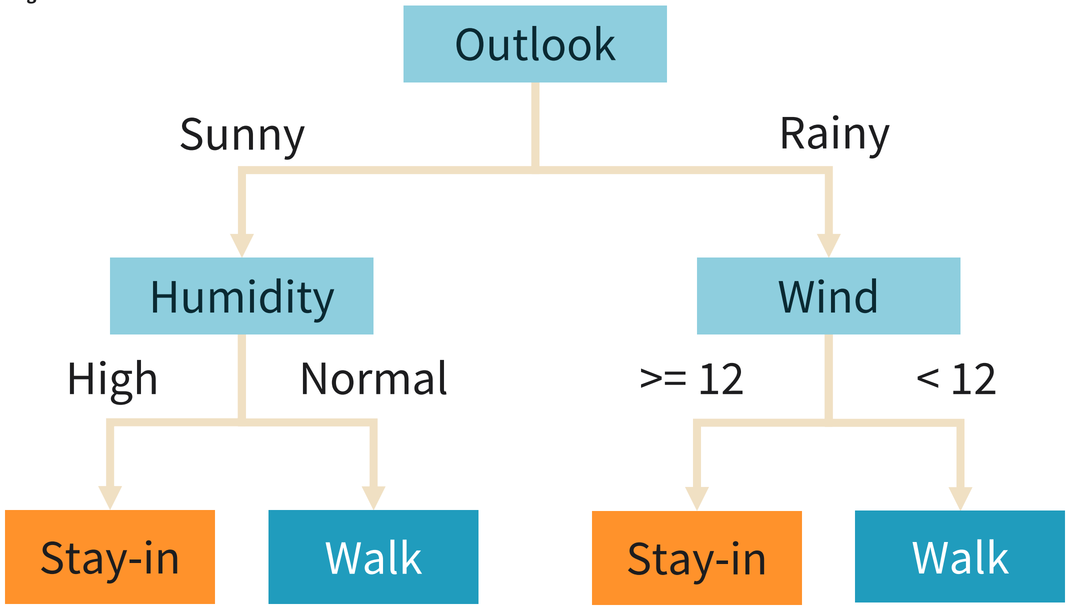
– Representation

- Each internal node specifies a test of some predictive attribute
- Each branch descending from a node corresponds to one of the possible values for this attribute
- Each leaf node assigns a class label

Tab. 1: Training data

Outlook	Humidity	Wind	Label
Rainy	High	20 km/h	Stay
Sunny	Normal	4 km/h	Walk
Sunny	Low	18 km/h	Walk
…	…	…	…

Img. 2: Decision tree



We can "translate" each path into IF-THEN rules (human readable)

Source of image 2: Christian Müller-Kett, 2021
Source of table 1: Christian Müller-Kett, 2021

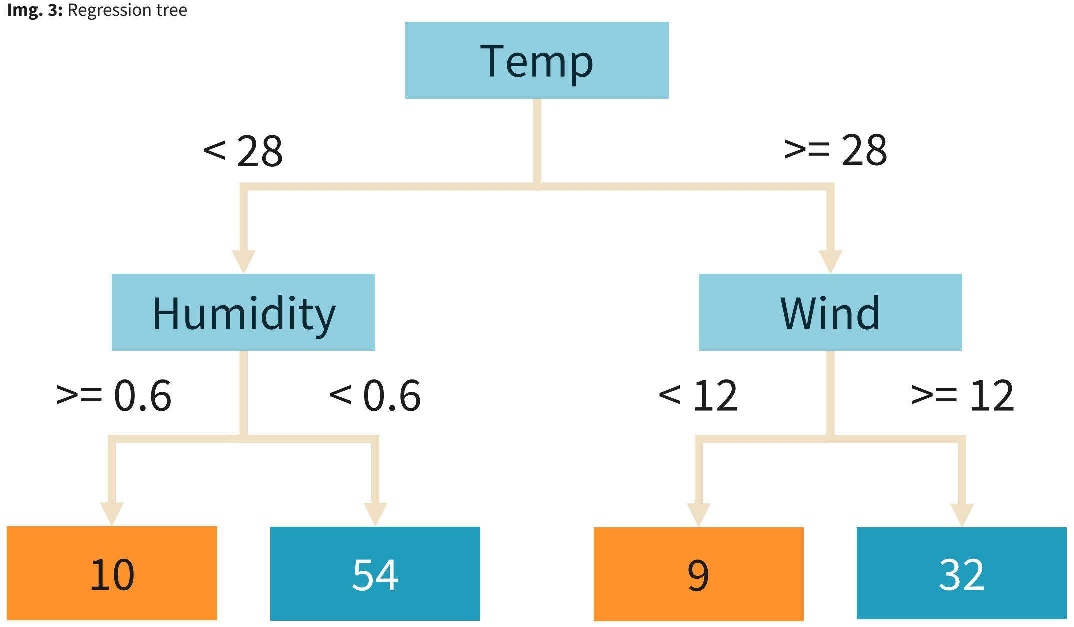
REGRESSION TREES

- Tree-based structures can also be used on numerical features and building regression models
- Numerical features become more manageable through a discretization process, i.e., by assigning threshold values
  - These numerical features can be treated like categorical features

Tab. 2: Training data

Temp	Humidity	Wind	Label
22	0.62	20 km/h	11
32	0.61	4 km/h	8
28	0.61	18 km/h	10
…	…	…	…

Img. 3: Regression tree



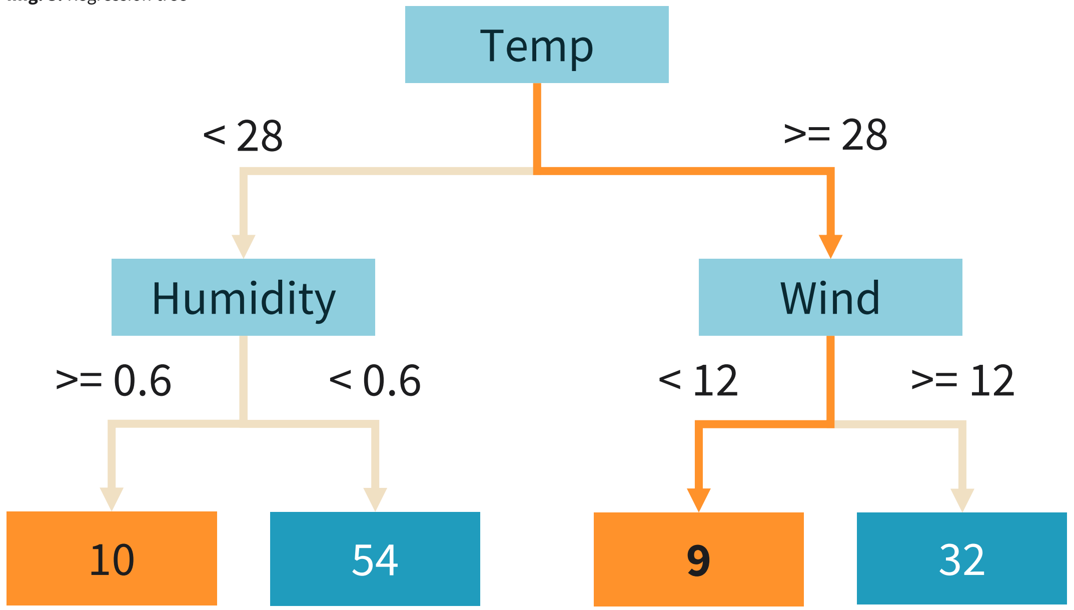
Source of image 3: Christian Müller-Kett, 2021
Source of table 2: Christian Müller-Kett, 2021

REGRESSION TREES

Tab. 2: Training data

Temp	Humidity	Wind	Label
22	0.62	20 km/h	11
32	0.61	4 km/h	8
28	0.61	18 km/h	10
…	…	…	…

Img. 3: Regression tree

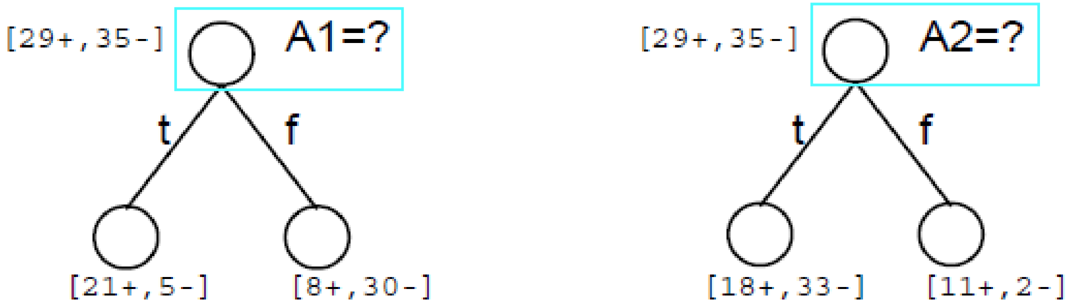


Source of image 3: Christian Müller-Kett, 2021
Source of table 2: Christian Müller-Kett, 2021

DECISION TREE - BASIC METHOD

- The tree is constructed in a top-down recursive divide-and-conquer manner
- At start, all the training examples are at the root node
- The question is "Which attribute should be tested/ selected for split?"
  - Attributes are evaluated using some statistical measure, which determines how well each attribute alone classifies the training examples.
  - The best attribute is selected and used as the splitting attribute at the root.
- For each possible value of the splitting attribute, a descendant of the root node is created and the instances are mapped to the appropriate descendant node.
- The procedure is repeated for each descendant node, so instances are partitioned recursively.
- "When do we stop partitioning?"
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning

- Which attribute to choose for splitting: A_1 or A_2?

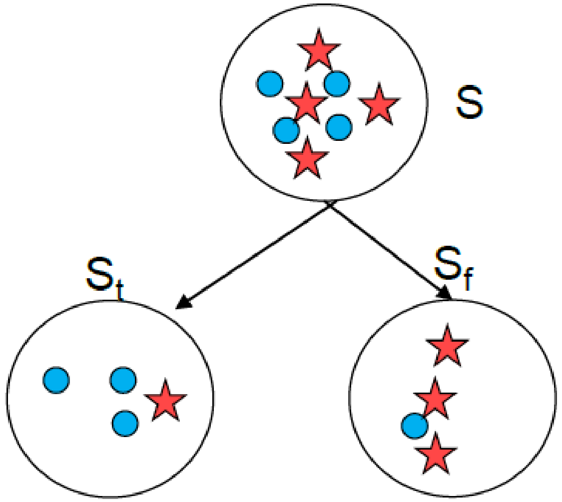


- Different split attribute selection measures
  - Information gain
  - Gini impurity (Gini index)
  - Sum of squared errors (SSE), with regards to regression tree

DECISION TREE – INFORMATION GAIN

- Used in ID3 (Quinlan, 1986)
- It uses entropy, a measure of pureness of the data
- The Information Gain *Gain(S, A)* of an attribute A relative to a collection of examples S measures the entropy reduction in S due to splitting on A:

$$G(S, A) = \underbrace{\boxed{\mathrm{Entropy}\left(S\right)}}_{\text{Before splitting}} - \underbrace{\boxed{\sum_{v \,\in\, \mathrm{values}(A)} \frac{|S_v|}{|S|} \mathrm{Entropy}\left(S_v\right)}}_{\text{After splitting on A}}$$



- Information Gain measures the expected reduction in entropy due to splitting on A
- The attribute with the higher entropy reduction is chosen for splitting

Source of the text: Ntoutsi, 2021

- Entropy comes from information theory.
  - It represents the average amount of information needed to identify the class label of an instance in S
  - The higher the entropy the more the information content
- Let S be a collection of positive and negative examples
  - p+: the percentage of positive examples in S
  - p-: the percentage of negative examples in S
- Entropy measures the impurity of S:

$$Entropy(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-)$$

in the general case
(k-classification problem)

$$Entropy(S) = \sum_{i=1}^{k} - p_i \log_2(p_i)$$

  - Entropy= 0, when all members belong to the same class
  - Entropy= 1, when there is an equal number of positive and negative examples

Source of the text: Ntoutsi, 2021

- What is the entropy in the following cases?
  - S: [9+,5-]
  - S: [7+,7-]
  - S: [14+,0-]

$$Entropy(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-)$$

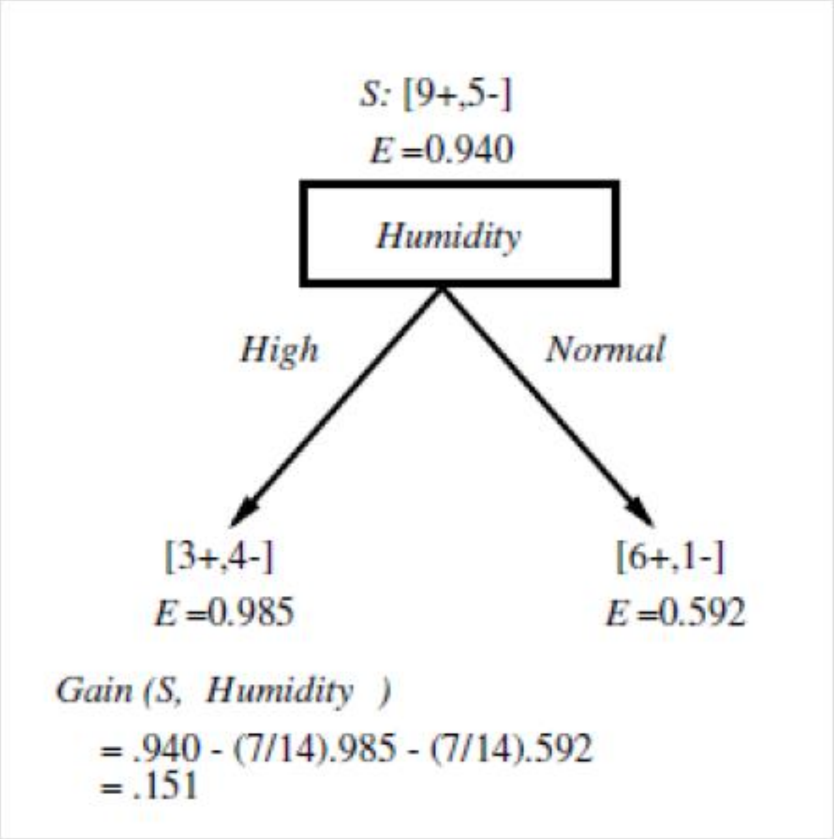
$$Entropy(S) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940$$

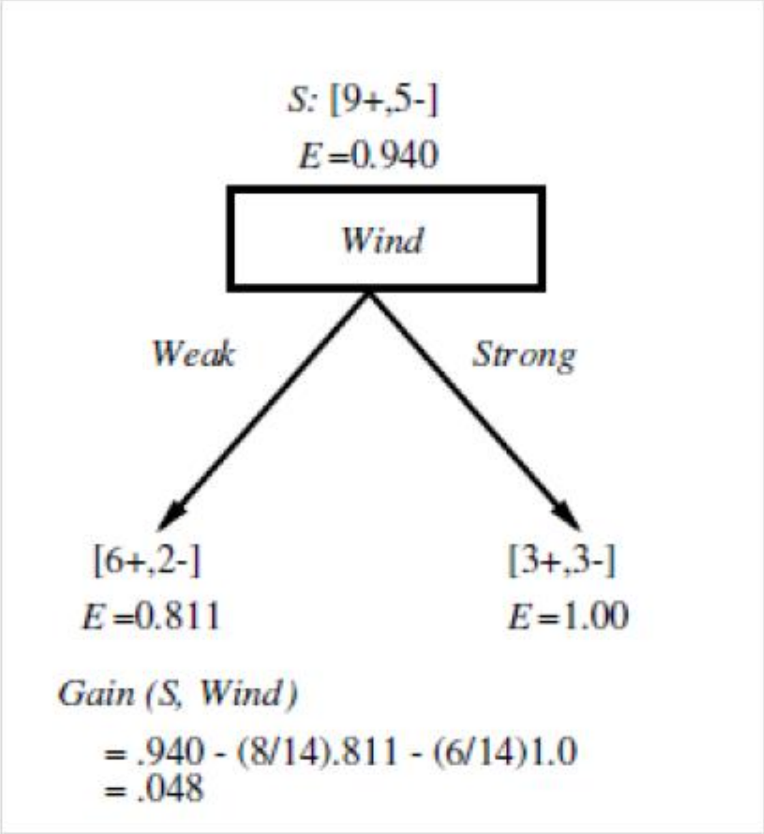
$$Entropy(S) = -\frac{7}{14}\log_2(\frac{7}{14}) - \frac{7}{14}\log_2(\frac{7}{14}) = 1$$

$$Entropy(S) = -\frac{14}{14}\log_2(\frac{14}{14}) - \frac{0}{14}\log_2(\frac{0}{14}) = 0$$

Source of the text: Ntoutsi, 2021

– Two options for splitting: “Humidity” and “Wind”?

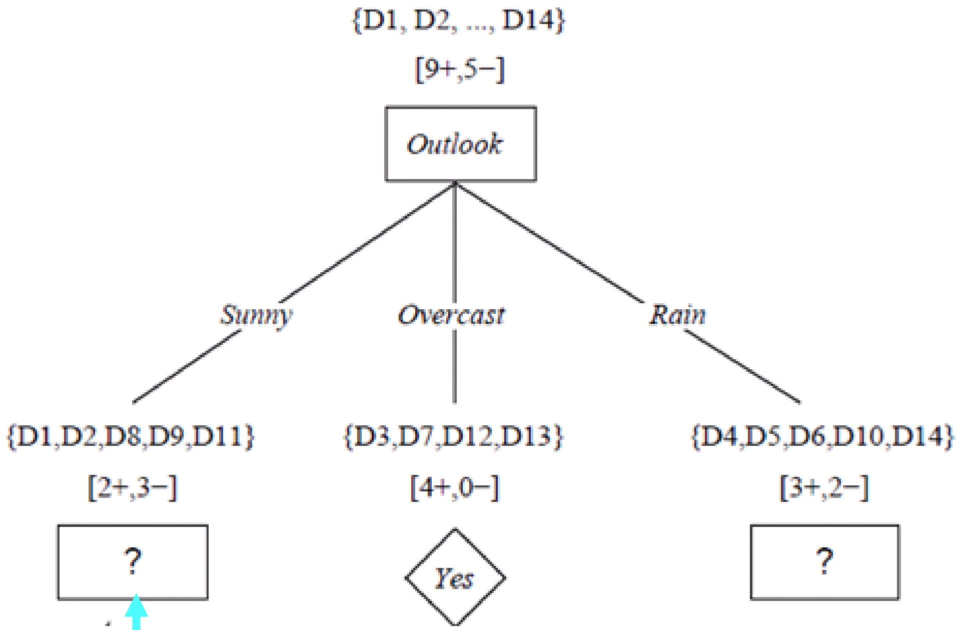




Source of the text: Ntoutsi, 2021

INFORMATION GAIN EXAMPLE

Repeat recursively



Which attribute should we choose for splitting here?

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

$$S_{sunny} = \{D1, D2, D8, D9, D11\}$$

$$Gain\,(S_{sunny}, Humidity) = .970 - (3/5)\,0.0 - (2/5)\,0.0 = .970$$

$$Gain\,(S_{sunny}, Temperature) = .970 - (2/5)\,0.0 - (2/5)\,1.0 - (1/5)\,0.0 = .570$$

$$Gain\,(S_{sunny}, Wind) = .970 - (2/5)\,1.0 - (3/5)\,.918 = .019$$

- Information gain is biased towards attributes with a large number of distinct values

$$G(S, A) = \text{Entropy}\left(S\right) - \sum_{v \in \text{values}(A)} \frac{|S_v|}{|S|} \text{Entropy}\left(S_v\right)$$

  - Consider unique identifiers like ID or credit card
  - Such attributes have a high information gain, because they uniquely identify each instance, but we do not want to include them in the decision tree
    - E.g., deciding how to treat a customer based on their credit card number is unlikely to generalize to customers we haven't seen before.
  - Measures have been proposed that “correct” this issue:
    - Gini impurity (Gini index)

GINI IMPURITY

- Used in CART (Breiman et al., 1984)
- Measure of impurity or divergence within a dataset
  - The probability of a randomly chosen observation to be misclassified
- Let a dataset S containing examples from k classes. Let p_i be the probability of class i in S. The Gini Index of S is given by: $Gini(S) = 1 - \sum_{i=1}^{k} p_i^2$
- Gini index considers a binary split for each attribute A. Let S is split based on A into two subsets S_1 and S_2.

$$Gini(S, A) = \frac{|S_1|}{|S|} Gini(S_1) + \frac{|S_2|}{|S|} Gini(S_2)$$

- We want to evaluate the reduction in the impurity of S based on A

$$\Delta Gini(S, A) = Gini(S) - Gini(S, A)$$

- The attribute A that provides the smallest Gini(S,A)(or the largest reduction in impurity) is chosen to split the node

Gini impurity

$$GI = 1 - \sum_{i=1}^{n} (p_i)^2$$

Example

- For a **sunny** outlook, there are **2 stay-in-days** and **4 walk** days in the training data
- $GI_{sunny} = 1 - \left(\frac{2}{2+4}\right)^2 + \left(\frac{4}{2+4}\right)^2 \approx 0.44$
- For a **rainy** outlook, there are **only stay-in-days** in the training data
- $GI_{rainy} = 1 - \left(\frac{4}{4+0}\right)^2 + \left(\frac{0}{4+0}\right)^2 = 0$
- $GI_{outlook} = \frac{GI_{sunny} + GI_{rainy}}{2} \approx 0.22$

Source of text: Boehmke & Greenwell, 2019, Breiman et al., 1984, Mitchell, 1997, Hastie et al., 2017

Information Gain (IG)

$$IG = H - \sum_{j=1}^{nA} p_j * H_j$$

Entropy (H)

$$H = -\sum_{i=1}^{n} p_i * \log(p_i)$$

Source of text: Boehmke & Greenwell, 2019, Breiman et al., 1984, Mitchell, 1997, Hastie et al., 2017

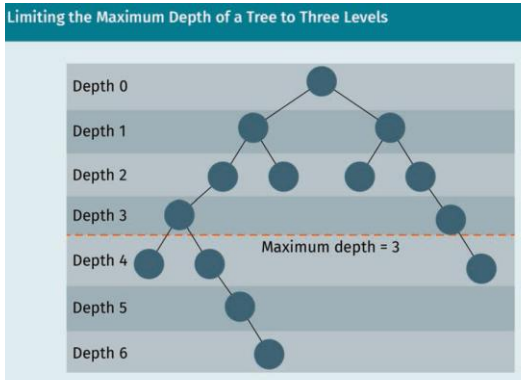
SUM OF SQUARED ERRORS (SSE)

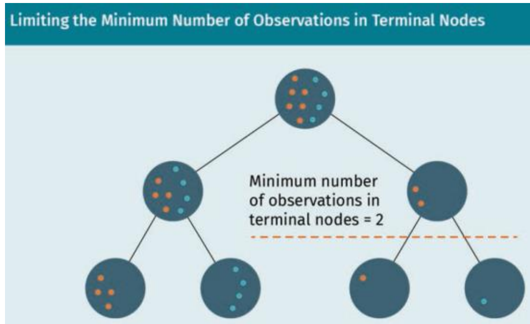
- Used in regression trees
- If we want to split a dataset S into two subsets S_1 and S_2,
  - Y: actual value, $\bar{y}$: mean value of the left/right side of the possible split

$$SSE = \sum_{i \in S_1} (y_i - \overline{y}_1)^2 + \sum_{i \in S_2} (y_i - \overline{y}_2)^2$$

- Finding the minimization of the SSE

- Risk of **overfitting**
- Stop criteria as **regularization** techniques
- **Maximum Depth**
  - Stop building decision trees after ***k* splits**
- **Minimum number of observations**
  - Stop building decision trees when there are less than ***n* samples in one node**
- **Pruning**
  - Calculate a **tree score**, minimizing the error and number of nodes
  - $C = Error + \lambda L$
  - Choose the tree with the best tree score

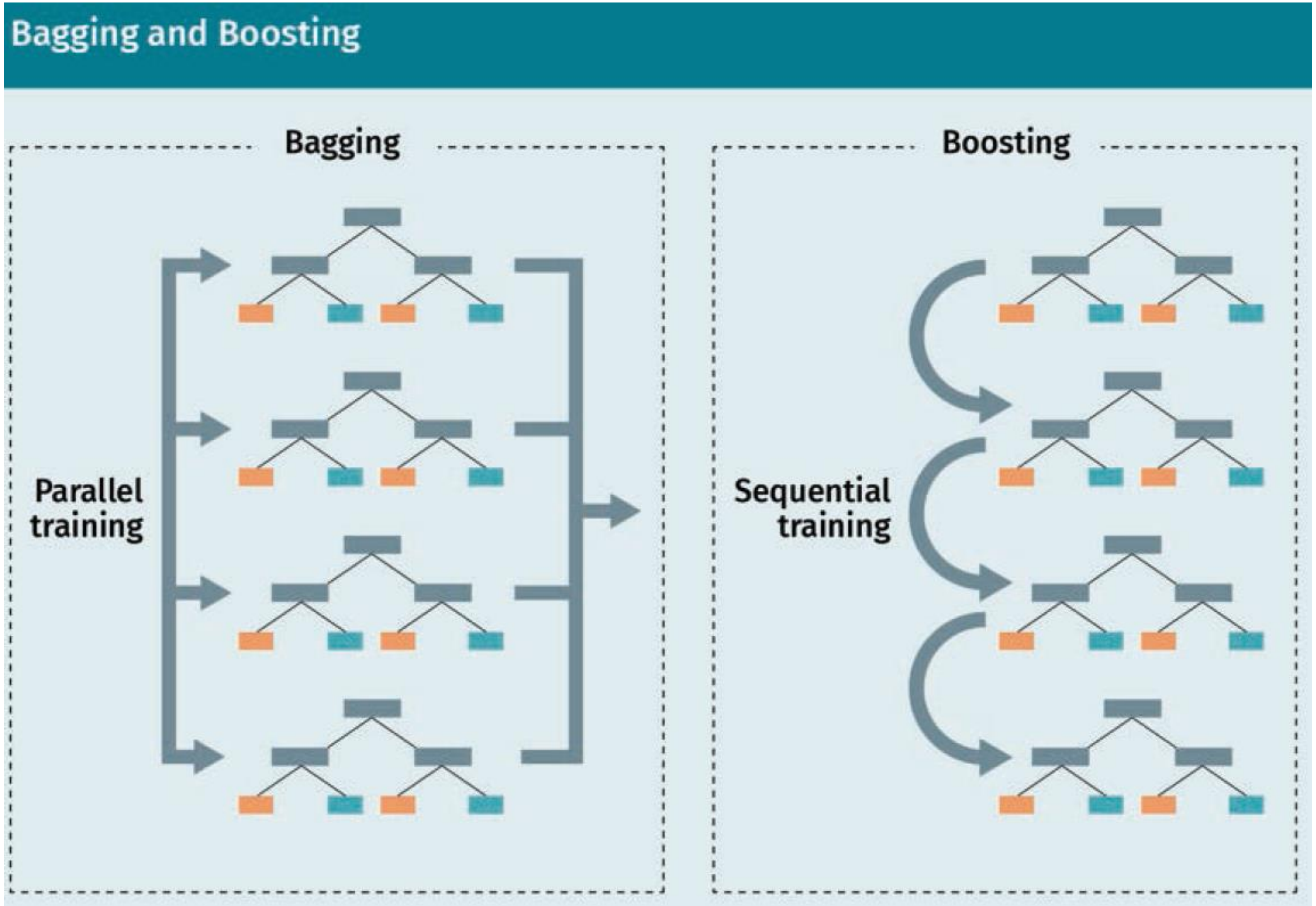




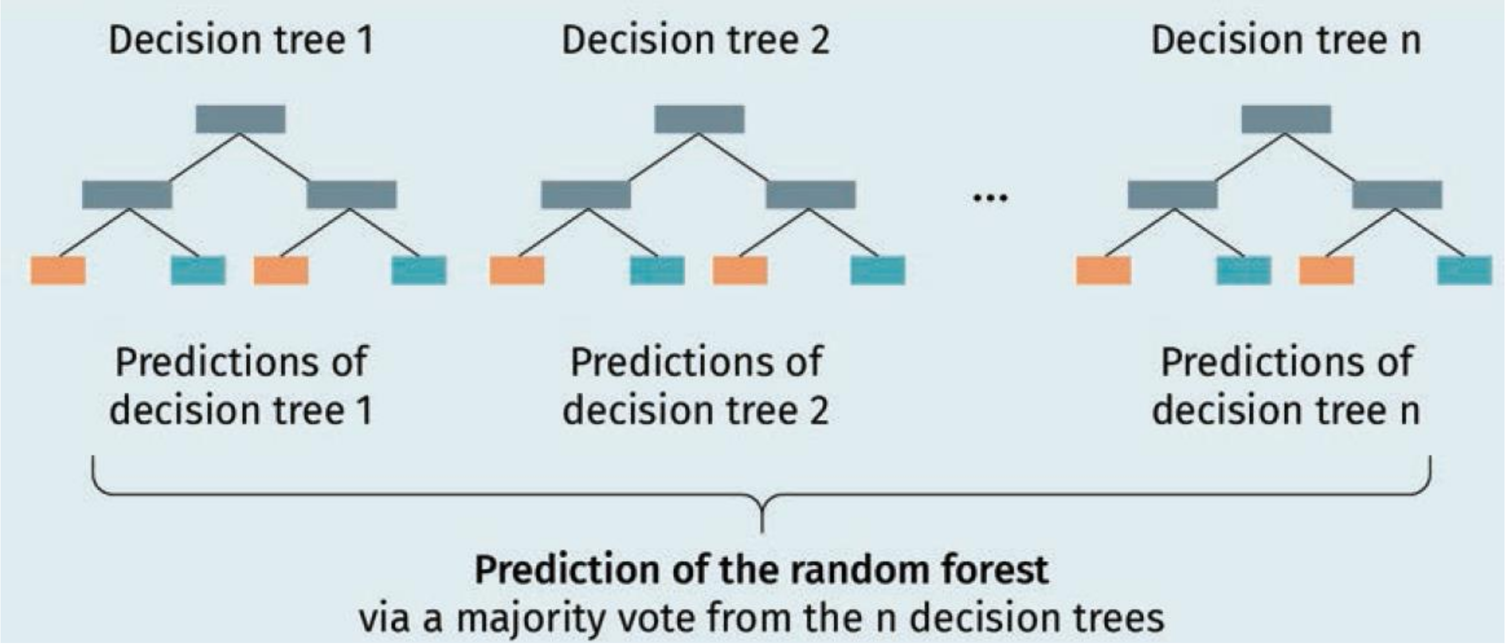
Source of text: Boehmke & Greenwell, 2019, Breiman et al., 1984, Mitchell, 1997

- **Combine weak estimators** to form **one strong estimator.**
- **Bagging**
  - Several decision trees trained by **varying data subsets.**
  - The final prediction as the **majority** or **average** of individual predictions.
- **Boosting**
  - Several decision trees trained **sequentially.**
  - Each tree is trained with a dataset **exaggerating** the **misclassified samples** from the previous tree.

Source of text: Boehmke & Greenwell, 2019, Breiman et al., 1984, Mitchell, 1997

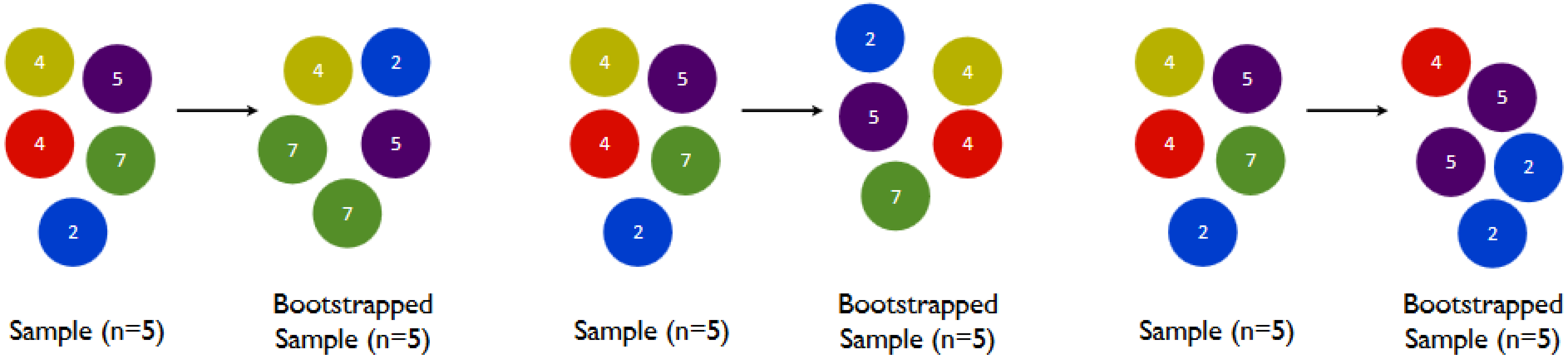

Bagging and Boosting
Bagging
Boosting
Parallel training
Sequential training

- The random forest (RF) is a high performing algorithm that bundles single decision trees into one strong estimator through bagging
- The final prediction of a RF is then determined by aggregating the predictions of the individual decision trees
  - Combining them via a majority vote



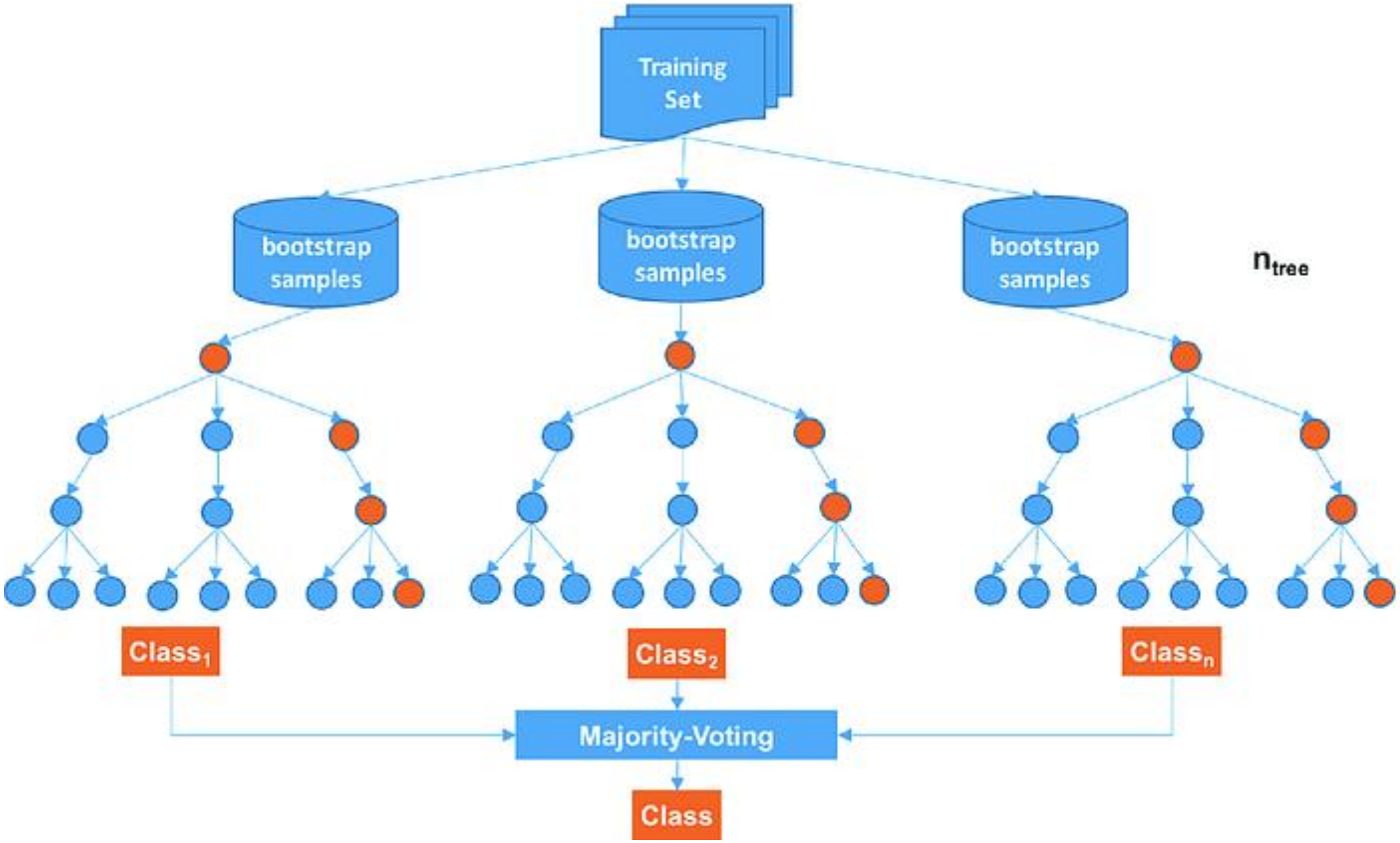
Source of image: Course book

- The individual decision trees are constructed independently of each other with the introduction of a random component
- This is done by building the trees on **bootstrap** copies of the training data
  - This procedure resamples a dataset to create many simulated samples of the same size by drawing randomly with laying back



Source of image: https://stats.stackexchange.com/questions/246726/size-of-bootstrap-samples

RANDOM FOREST



Source of image: https://medium.com/@uma.bollikonda/random-forest-machine-learning-algorithm-401bdcd7a0b8

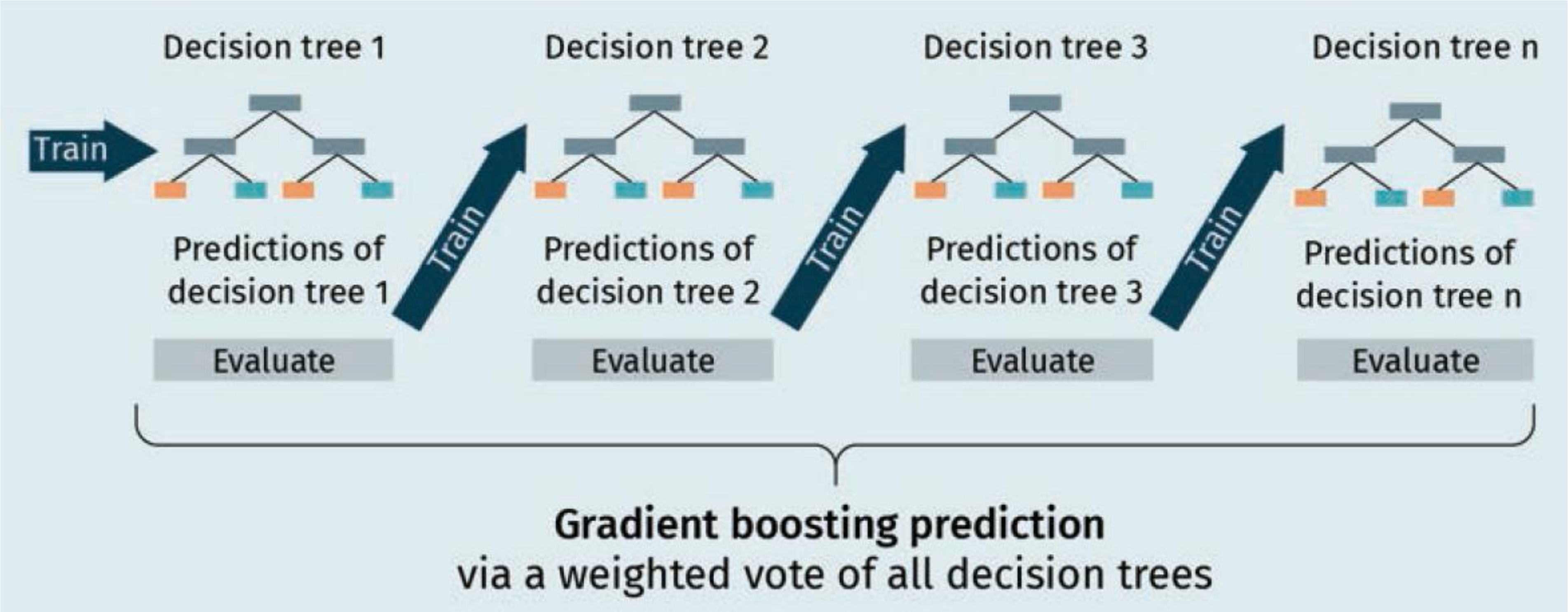
- Select the number of trees to construct (hyperparameter "**n_trees**")
- For all number of trees:
  - Generate a bootstrap sample of the original data.
  - Grow a regression/classification tree based on the bootstrapped data.
  - For each node/split, perform the following set of actions:
    - Select a number "**m_try**" of features at random from all p features.
    - Pick the best feature/split-point among the "**m_try**" tested features.
    - Split the node into two child nodes.
  - Use common tree model stop criteria to determine when a tree is completed and unpruned.
  - Output the ensemble of trees.
  - Let each of the trees make a prediction.
  - Use the individual predictions in a voting process in which the final prediction is determined.

1. **Bootstrapping**
    a. Randomly draw a subset of the samples (**in the bag**)
    b. withhold the rest (**out-of-bag**)
    c. Fill the bag with sample duplicates
2. **Random subspace**
    a) Randomly draw features
3. **Train a decision tree**
    a) Use the out-of-bag samples to calculate the out-of-bag error
4. **Repeat** steps 1-3 for *n* trees
5. Obtain **predictions** for unseen data from each tree
6. The final prediction is the **majority/average** of individual predictions

Source of text: Boehmke & Greenwell, 2019, Breiman et al., 1984, Mitchell, 1997

- Gradient boosting uses a large number of individual weak estimators (usually decision trees) that are combined into one strong estimator.
- The individual estimators are not trained in parallel and independently of each other
- They are trained sequentially, and one estimator helps optimize the next
- The misclassified samples in one iteration are exaggerated in the following iteration, thereby placing emphasis on the need to classify these samples correctly in the next decision tree to be trained

1. A loss function we want to optimize
2. A set of weak learners generating predictions
3. An additive model for combining the predictions of the weak learners into one strong predictor (thereby minimizing the loss function)



Source of the image: Course book

- Select the number of trees to construct (hyperparameter “**n_trees**”).
- Fit the first weak estimator to the given training data X and generate predictions ŷ, i.e., $F_1(x)=\hat{y}$
- Fit the next weak estimator to the residuals of the previous one, i.e, $h_1(x)=y-F_1(x)$
- Add this weak estimator to the model, i.e., $F_2(x)=F_1(x)+h_1(x)$
- Fit the next weak estimator to the residuals of F_2: $h_2(x)=y-F_2(x)$
- Add this weak estimator to the model, i.e., $F_3(x)=F_2(x)+h_2(x)$
- Continue this process until the number of prospective trees has been reached.

The resulting strong estimator can be mathematically expressed as the additive combination of the single *i* weak estimators of number *n*:

$$f(x) = \sum_{i=1}^{n} f^i(\mathrm{x})$$

1. Select ***n* trees** to be constructed.
2. Train a **weak estimator.**
3. **Fit another weak estimator** to the **residuals** of the previous one.
4. Add the new estimator to the model (**linear combination**).
5. **Repeat** steps 2-4.
6. Obtain the final prediction as a **linear combination of all estimators.**

Source of text: Boehmke & Greenwell, 2019, Breiman et al., 1984, Mitchell, 1997

REVIEW STUDY GOALS

- Explain the concept of **decision** and **regression trees**.
- Explain the power of the **ensemble methods** widely used in practice.
- Define **bagging** and **boosting**.
- Apply two very popular ensemble models on your own with the use of **Python**.

SESSION 5

TRANSFER TASK

A start-up that sells **sustainable products in smaller stores** has been very successful in recent years. As a result, more stores are to be opened worldwide.

As a Data Scientist, you and your team are tasked with training a **machine learning model predicting product demand** one week ahead.

Explain why tree-based machine learning models can be used for this, although it is a **regression** and not a **classification** task.

Discuss ways to **avoid overfitting** the data.

You do not dispose of a very large machine with multiple CPU cores. Instead, your **laptop with only two cores** must do the job. Discuss **computational considerations** in training a **bagging or boosting** ensemble model.

Please present your results.
The results will be discussed in plenary.

1. The estimators typically used in gradient boosting are called …
   a) … decision trees.
   b) … support vector machines.
   c) … logistic regressions.
   d) … linear regressions.

2. What is the umbrella term for algorithms such as random forest and gradient boosting?
   a) bagging methods
   b) boosting methods
   c) strong estimators
   d) ensemble methods

3. Which of the following is not a typical hyperparameter of a random forest?
   a) number of estimators
   b) number of coefficients
   c) maximum depth
   d) minimum samples in leave nodes

LIST OF SOURCES

Boehmke, B., & Greenwell, B. (2019). *Hands-on machine learning with R*. Chapman & Hall.

Breiman, L., Friedman, J., Olshen, R. A., & Stone, J. S. (1984). *Classification and regression trees*. Chapman & Hall. https://doi.org/10.1201/9781315139470

Hastie, T., Tibshirani, R., Friedman, J. H. (2017). *The elements of statistical learning. Data mining, inference, and prediction*. Second edition. New York, NY: Springer.

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.