CSI5180. Machine Learning for Bioinformatics Applications

Decision Trees

by

Marcel Turcotte

Version **November 6, 2019**

Preamble

Preamble

Decision Trees

In this lecture, we discuss decision trees. One of their main advantages is interpretability. Also, they naturally work with a mixture of data (feature) types. When used in the context of ensemble learning, their performance is generally excellent.

General objective :

- **Explain** what decision trees are, how they are built, and how they can be used to classify data.

Learning objectives

- **Explain** what decision trees are.
- **Decribe** the algorithm to construct a decision tree.
- **Discuss** the concept of purity.
- **Explain** how to use decision trees to classify new examples.

Reading:

- Carl Kingsford and Steven L Salzberg. What are decision trees? *Nat Biotechnol* **26**(9):10113, Sep 2008.
- Pierre Geurts, Alexandre Irrthum, and Louis Wehenkel. Supervised learning with decision tree-based methods in computational and systems biology. *Mol Biosyst* **5**(12):1593605, Dec 2009.

Plan

1. Preamble

2. Introduction

3. What is a decision tree?

4. Construction

5. Limitation

6. Random Forest

7. Applications

8. Prologue

Introduction

Decision Trees

- **Decision trees** are **supervised** learning algorithms.

Decision Trees

- **Decision trees** are **supervised** learning algorithms.
- **Tasks**:

Decision Trees

- **Decision trees** are **supervised** learning algorithms.
- **Tasks**:
    - **Regression:** y_i is a real value (not discussed here);

Decision Trees

- **Decision trees** are **supervised** learning algorithms.
- **Tasks**:
  - **Regression:** y_i is a real value (not discussed here);
  - **Classification:** y_i is a class.

Decision Trees

- **Decision trees** are **supervised** learning algorithms.
- **Tasks**:
  - **Regression:** y_i is a real value (not discussed here);
  - **Classification:** y_i is a class.
- At the heart of **Random Forest** algorithms, which are some of the **best** algorithms when the **number of available examples is low**.

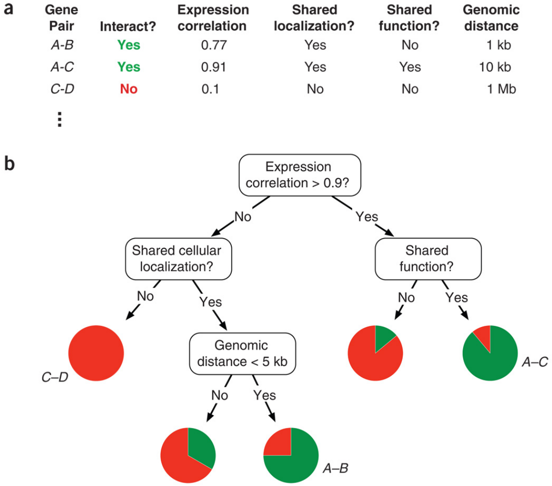
Decision Trees

- **Decision trees** are **supervised** learning algorithms.
- **Tasks**:
  - **Regression:** y_i is a real value (not discussed here);
  - **Classification:** y_i is a class.
- At the heart of **Random Forest** algorithms, which are some of the **best** algorithms when the **number of available examples is low**.
- Handles a mixture of **categorical** and **real-valued** features, as well as **missing values**, depending on the implementation (Scikit-Learn's implementation does not handle the missing values).

Decision Trees

- **Decision trees** are **supervised** learning algorithms.
- **Tasks**:
  - **Regression:** y_i is a real value (not discussed here);
  - **Classification:** y_i is a class.
- At the heart of **Random Forest** algorithms, which are some of the **best** algorithms when the **number of available examples is low**.
- Handles a mixture of **categorical** and **real-valued** features, as well as **missing values**, depending on the implementation (Scikit-Learn's implementation does not handle the missing values).
- The resulting **model** is easily **interpretable** by humans!

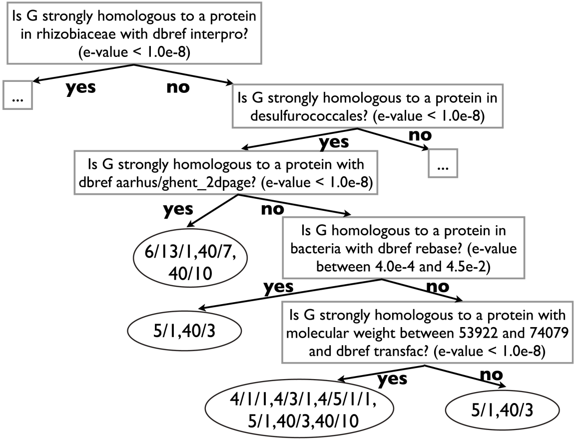
Interpretable models



Gene Pair	Interact?	Expression correlation	Shared localization?	Shared function?	Genomic distance
A-B	Yes	0.77	Yes	No	1 kb
A-C	Yes	0.91	Yes	Yes	10 kb
C-D	No	0.1	No	No	1 Mb
⋮					

C. Kingsford and S. L. Salzberg. What are decision trees? *Nat Biotechnol* **26**(9):10113, 2008.

Interpretable models



Is G strongly homologous to a protein in rhizobiaceae with dbref interpro? (e-value < 1.0e-8)

yes

...

no

Is G strongly homologous to a protein in desulfurococcales? (e-value < 1.0e-8)

yes

no

...

Is G strongly homologous to a protein with dbref aarhus/ghent_2dpage? (e-value < 1.0e-8)

yes

6/13/1,40/7, 40/10

no

Is G homologous to a protein in bacteria with dbref rebase? (e-value between 4.0e-4 and 4.5e-2)

yes

5/1,40/3

no

Is G strongly homologous to a protein with molecular weight between 53922 and 74079 and dbref transfac? (e-value < 1.0e-8)

yes

4/1/1,4/3/1,4/5/1/1, 5/1,40/3,40/10

no

5/1,40/3

Schietgat, L. et al. Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC bioinformatics* **11**, 14 (2010).

What is a decision tree?

What is a decision tree?

- A **decision tree** is a **hierarchical structure** (directed acyclic graph) used to make decisions (classification, regression).

What is a decision tree?

- A **decision tree** is a **hierarchical structure** (directed acyclic graph) used to make decisions (classification, regression).
- Each **internal node** is a (binary) test for some feature j.

What is a decision tree?

- A **decision tree** is a **hierarchical structure** (directed acyclic graph) used to make decisions (classification, regression).
- Each **internal node** is a (binary) test for some feature j.
    - E.g. is the **expression of a given gene in the given sample** higher than some threshold.

What is a decision tree?

- A **decision tree** is a **hierarchical structure** (directed acyclic graph) used to make decisions (classification, regression).
- Each **internal node** is a (binary) test for some feature j.
  - E.g. is the **expression of a given gene in the given sample** higher than some threshold.
- **Leaves** are decision nodes.

What is a decision tree?

- A **decision tree** is a **hierarchical structure** (directed acyclic graph) used to make decisions (classification, regression).
- Each **internal node** is a (binary) test for some feature j.
    - E.g. is the **expression of a given gene in the given sample** higher than some threshold.
- **Leaves** are decision nodes.
- The **“structure”** of the tree is **learnt** from the **training data**.

What is a decision tree?

- A **decision tree** is a **hierarchical structure** (directed acyclic graph) used to make decisions (classification, regression).
- Each **internal node** is a (binary) test for some feature j.
  - E.g. is the **expression of a given gene in the given sample** higher than some threshold.
- **Leaves** are decision nodes.
- The **“structure”** of the tree is **learnt** from the **training data**.

What is a decision tree?

- A **decision tree** is a **hierarchical structure** (directed acyclic graph) used to make decisions (classification, regression).
- Each **internal node** is a (binary) test for some feature j.
  - E.g. is the **expression of a given gene in the given sample** higher than some threshold.
- **Leaves** are decision nodes.
- The **"structure"** of the tree is **learnt** from the **training data**.

$\Rightarrow$ More complex representations are possible (for instance, ID3 allows for more than two children).

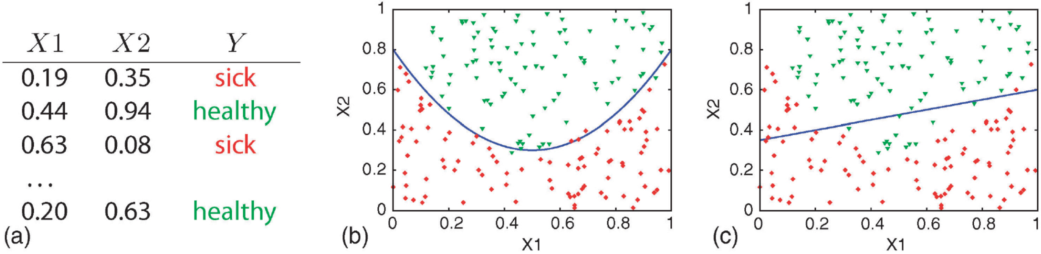
Classifying new examples

- It suffices to start at the **root** of the tree, answering a series of **binary questions**, until **reaching a leave**, the value found in that leave is the **label** of that example.

Classifying new examples

- It suffices to start at the **root** of the tree, answering a series of **binary questions**, until **reaching a leave**, the value found in that leave is the **label** of that example.
    - In some cases (algorithms), a leave contains a **probability distribution** — the proportion of the training data belonging to class k, for all values of k, in this leaf.

Simple decision boundary

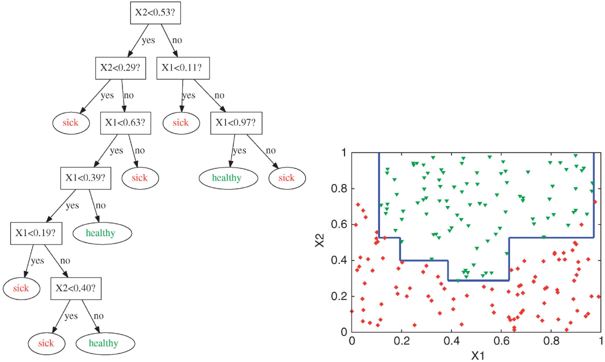


$X1$	$X2$	Y
0.19	0.35	sick
0.44	0.94	healthy
0.63	0.08	sick
…		
0.20	0.63	healthy

- **Logistic Regression** has a simple **decision boundary**

Source: Geurts, P., Irrthum, A. & Wehenkel, L. Supervised learning with decision tree-based methods in computational and systems biology. *Mol Biosyst* **5** 15931605 (2009).

Complex decision boundary



- **Decision trees** can produce a complex and irregular **decision boundary**

Source: Geurts, P., Irrthum, A. & Wehenkel, L. Supervised learning with decision tree-based methods in computational and systems biology. *Mol Biosyst* **5** 15931605 (2009).

Construction

Building a decision tree

- **Incremental process:** starting with an **empty tree**, **nodes are added one by one** guided by the **training data** until the data is **perfectly classified** (or some other criteria applies, e.g. maximum depth).

Building a decision tree

- **Incremental process:** starting with an **empty tree**, **nodes are added one by one** guided by the **training data** until the data is **perfectly classified** (or some other criteria applies, e.g. maximum depth).
- Consider the case of **creating the first node**.

Building a decision tree

- **Incremental process:** starting with an **empty tree**, **nodes are added one by one** guided by the **training data** until the data is **perfectly classified** (or some other criteria applies, e.g. maximum depth).
- Consider the case of **creating the first node**.
  - It makes sense to **consider all** D **features** to create a test. For **each feature**, one can consider several thresholds (based on the **observed values** in our **training set**).

Building a decision tree

- **Incremental process:** starting with an **empty tree**, **nodes are added one by one** guided by the **training data** until the data is **perfectly classified** (or some other criteria applies, e.g. maximum depth).
- Consider the case of **creating the first node**.
  - It makes sense to **consider all** D **features** to create a test. For **each feature**, one can consider several thresholds (based on the **observed values** in our **training set**).
  - **What would be good criteria to select the best candidate?**

Building a decision tree

- **Incremental process:** starting with an **empty tree**, **nodes are added one by one** guided by the **training data** until the data is **perfectly classified** (or some other criteria applies, e.g. maximum depth).
- Consider the case of **creating the first node**.
    - It makes sense to **consider all** D **features** to create a test. For **each feature**, one can consider several thresholds (based on the **observed values** in our **training set**).
    - **What would be good criteria to select the best candidate?**
        - **Heterogeneity** (**impurity**) and **homogeneity** of the resulting sets.

Building a decision tree

- **Incremental process:** starting with an **empty tree**, **nodes are added one by one** guided by the **training data** until the data is **perfectly classified** (or some other criteria applies, e.g. maximum depth).
- Consider the case of **creating the first node**.
  - It makes sense to **consider all** D **features** to create a test. For **each feature**, one can consider several thresholds (based on the **observed values** in our **training set**).
  - **What would be good criteria to select the best candidate?**
    - **Heterogeneity** (**impurity**) and **homogeneity** of the resulting sets.
    - Ideally, **each one of the two resulting sets** would contain data from a **single class**!

Building a decision tree

- **Incremental process:** starting with an **empty tree**, **nodes are added one by one** guided by the **training data** until the data is **perfectly classified** (or some other criteria applies, e.g. maximum depth).
- Consider the case of **creating the first node**.
    - It makes sense to **consider all** D **features** to create a test. For **each feature**, one can consider several thresholds (based on the **observed values** in our **training set**).
    - **What would be good criteria to select the best candidate?**
        - **Heterogeneity** (**impurity**) and **homogeneity** of the resulting sets.
        - Ideally, **each one of the two resulting sets** would contain data from a **single class**!
        - **Entropy** and **Geni index** are two popular choices.

Objective function

- Objective function for **sklearn.tree.DecisionTreeClassifier** (CART):

$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

Objective function

- Objective function for **sklearn.tree.DecisionTreeClassifier** (CART):

$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

- The cost of partitioning the data using **feature** k and **threshold** t_k.

Objective function

- Objective function for **sklearn.tree.DecisionTreeClassifier** (CART):

$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

- The cost of partitioning the data using **feature** k and **threshold** t_k.
- m_{left} and m_{right} is the **number of examples** in the **left** and **right** subsets, respectively, and m is the number of examples before splitting the data.

Objective function

- Objective function for **sklearn.tree.DecisionTreeClassifier** (CART):

$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

- The cost of partitioning the data using **feature** k and **threshold** t_k.
- m_{left} and m_{right} is the **number of examples** in the **left** and **right** subsets, respectively, and m is the number of examples before splitting the data.
- G_{left} and G_{right} is the **impurity** of the **left** and **right** subsets, respectively.

Geni index

- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

Geni index

- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

- $p_{i,k}$ is the proportion of the examples from this class k in the node i.

Geni index

- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

- $p_{i,k}$ is the proportion of the examples from this class k in the node i.
- **Examples**:

Geni index

- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

- $p_{i,k}$ is the proportion of the examples from this class k in the node i.
- **Examples**:
  - $1 - (0/100)^2 + (100/100)^2 = 0$ (pure)

Geni index

- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

- $p_{i,k}$ is the proportion of the examples from this class k in the node i.
- **Examples**:
  - $1 - (0/100)^2 + (100/100)^2 = 0$ (pure)
  - $1 - (25/100)^2 + (75/100)^2 = 0.375$

Geni index

- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

- $p_{i,k}$ is the proportion of the examples from this class k in the node i.
- **Examples**:
  - $1 - (0/100)^2 + (100/100)^2 = 0$ (pure)
  - $1 - (25/100)^2 + (75/100)^2 = 0.375$
  - $1 - (50/100)^2 + (50/100)^2 = 0.5$

Geni index

- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

- $p_{i,k}$ is the proportion of the examples from this class k in the node i.
- **Examples**:
  - $1 - (0/100)^2 + (100/100)^2 = 0$ (pure)
  - $1 - (25/100)^2 + (75/100)^2 = 0.375$
  - $1 - (50/100)^2 + (50/100)^2 = 0.5$
  - $1 - 10 \times (10/100)^2 = 0.9$

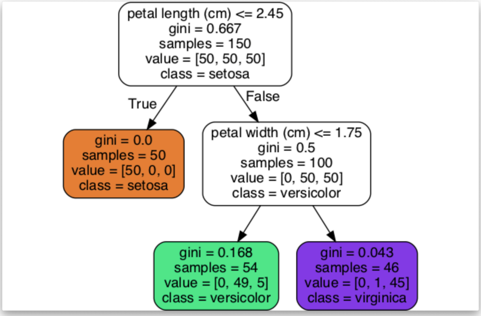
Geni index

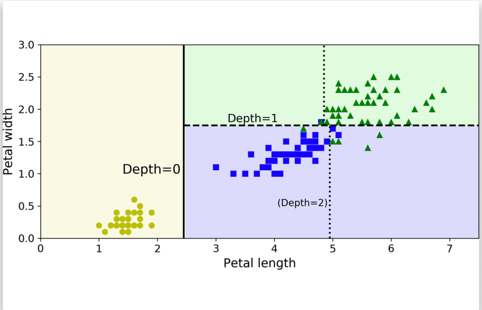
- **Gini index** (default)

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$$

- $p_{i,k}$ is the proportion of the examples from this class k in the node i.
- **Examples**:
  - $1 - (0/100)^2 + (100/100)^2 = 0$ (pure)
  - $1 - (25/100)^2 + (75/100)^2 = 0.375$
  - $1 - (50/100)^2 + (50/100)^2 = 0.5$
  - $1 - 10 \times (10/100)^2 = 0.9$
  - $1 - 100 \times (1/100)^2 = 0.99$

Iris data set





Source: [6] Figures 6.1 and 6.2

Entropy (or uncertainty)

- Given a set of training examples, let p_k be **proportion of these examples belonging to the class** C_k for $k \in \{1, 2, \ldots, m\}$ (a probability distribution).

Entropy (or uncertainty)

- Given a set of training examples, let p_k be **proportion of these examples belonging to the class** C_k for $k \in \{1, 2, \ldots, m\}$ (a probability distribution).
- **Entropy** is defined as follows:

$$H = -\sum_{k=1}^{m} p_k \log_2 p_k$$

Entropy (or uncertainty)

- Given a set of training examples, let p_k be **proportion of these examples belonging to the class** C_k for $k \in \{1, 2, \ldots, m\}$ (a probability distribution).
- **Entropy** is defined as follows:

$$H = -\sum_{k=1}^{m} p_k \log_2 p_k$$

- Entropy is **maximum** when the events are all **equiprobable**. The maximum value is $\log_2 m$.

Entropy (or uncertainty)

- Given a set of training examples, let p_k be **proportion of these examples belonging to the class** C_k for $k \in \{1, 2, \dots, m\}$ (a probability distribution).
- **Entropy** is defined as follows:

$$H = -\sum_{k=1}^{m} p_k \log_2 p_k$$

- Entropy is **maximum** when the events are all **equiprobable**. The maximum value is $\log_2 m$.
- The entropy approaches **0** as the probability of one class, p_k, approches **1**.

Entropy (or uncertainty)

- Given a set of training examples, let p_k be **proportion of these examples belonging to the class** C_k for $k \in \{1, 2, \ldots, m\}$ (a probability distribution).
- **Entropy** is defined as follows:

$$H = -\sum_{k=1}^{m} p_k \log_2 p_k$$

- Entropy is **maximum** when the events are all **equiprobable**. The maximum value is $\log_2 m$.
- The entropy approaches **0** as the probability of one class, p_k, approches **1**.
- **Entropy** and **Gini** produce similar results:

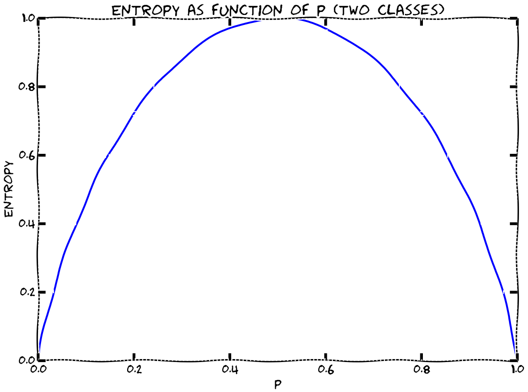
Entropy (or uncertainty)

- Given a set of training examples, let p_k be **proportion of these examples belonging to the class** C_k for $k \in \{1, 2, \ldots, m\}$ (a probability distribution).
- **Entropy** is defined as follows:

$$H = -\sum_{k=1}^{m} p_k \log_2 p_k$$

- Entropy is **maximum** when the events are all **equiprobable**. The maximum value is $\log_2 m$.
- The entropy approaches **0** as the probability of one class, p_k, approches **1**.
- **Entropy** and **Gini** produce similar results:
  - `https://sebastianraschka.com/faq/docs/decision-tree-binary.html`

Entropy (uncertainty) for two classes



- Consider a **probability space** with two **outcomes**.

$$H = -[(1-p)\log_2(1-p) + p\log_2 p]$$

Entropy (uncertainty) for two classes

```python
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

def lg2(v):
    return np.where(v != 0.0, np.log2(v), 0.0)

with plt.xkcd():
    p = np.arange(0.0, 1.0, 0.001)
    h = -((1-p) * lg2(1-p) + p*lg2(p))
    fig, ax = plt.subplots()
    ax.plot(p, h)
    ax.set(xlabel='p', ylabel='Entropy', title='Entropy as...')
    ax.grid()
    plt.show()
```

Stopping criteria

- All the examples in a given node belong to the **same class**.

Stopping criteria

- All the examples in a given node belong to the **same class**.
- Depth of the tree would exceed **max_depth**.

Stopping criteria

- All the examples in a given node belong to the **same class**.
- Depth of the tree would exceed **max_depth**.
- Number of examples in the node is **min_sample_split** or less.

Stopping criteria

- All the examples in a given node belong to the **same class**.
- Depth of the tree would exceed **max_depth**.
- Number of examples in the node is **min_sample_split** or less.
- None of the splits decreases impurity sufficiently (**min_impurity_decrease**).

Stopping criteria

- All the examples in a given node belong to the **same class**.
- Depth of the tree would exceed **max_depth**.
- Number of examples in the node is **min_sample_split** or less.
- None of the splits decreases impurity sufficiently (**min_impurity_decrease**).
- See documentation for other criteria.

Limitation

Limitations

- Possibly creates **large trees**

Limitations

- Possibly creates **large trees**
    - Challenge for **interpretation**

Limitations

- Possibly creates **large trees**
    - Challenge for **interpretation**
    - **Overfitting**

Limitations

- Possibly creates **large trees**
  - Challenge for **interpretation**
  - **Overfitting**
- **Greedy algorithm**, no guarantee to find the optimal tree.

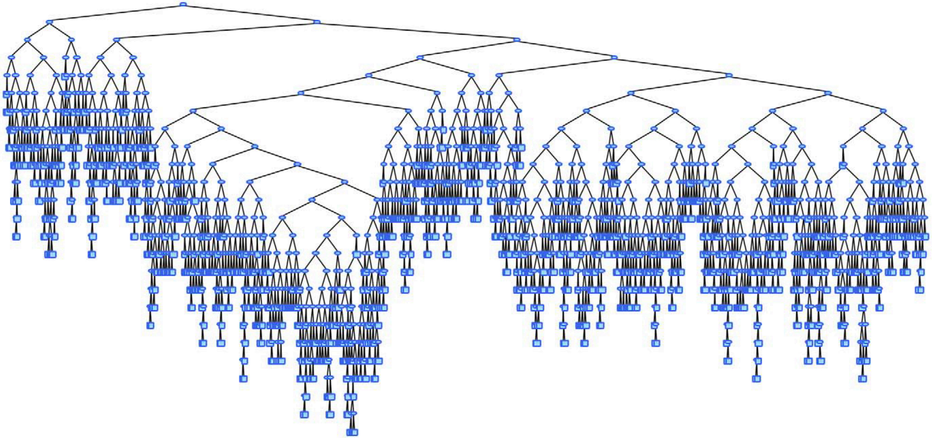
Limitations

- Possibly creates **large trees**
  - Challenge for **interpretation**
  - **Overfitting**
- **Greedy algorithm**, no guarantee to find the optimal tree.
- **Small changes** to the data set, produces **vastly different trees**.

Limitations

- Possibly creates **large trees**
  - Challenge for **interpretation**
  - **Overfitting**
- **Greedy algorithm**, no guarantee to find the optimal tree.
- **Small changes** to the data set, produces **vastly different trees**.
- The **decision boundaries** are orthogonal, which makes then **sensitive to rotations** (this can be alleviated by first running PCA [Principal Component Analysis], then applying decision trees on the transformed data).

Large trees



Stiglic, G., Kocbek, S., Pernek, I. & Kokol, P. Comprehensive Decision Tree Models in Bioinformatics. *PLoS ONE* **7**, (2012).

Regularization

- Limiting the **maximum depth** of the tree is a form of **regularization**.

Regularization

- Limiting the **maximum depth** of the tree is a form of **regularization**.
  - Likewise, the values for the other parameters, such as **min_impurity_decrease**, can be determined using a **validation set**.

Regularization

- Limiting the **maximum depth** of the tree is a form of **regularization**.
  - Likewise, the values for the other parameters, such as **min_impurity_decrease**, can be determined using a **validation set**.
- Another **regularization** technique for decision trees is known as **pruning**.

Regularization

- Limiting the **maximum depth** of the tree is a form of **regularization**.
  - Likewise, the values for the other parameters, such as **min_impurity_decrease**, can be determined using a **validation set**.
- Another **regularization** technique for decision trees is known as **pruning**.
  - In a bottom-up fashion, **nodes are removed** if this reduces the **classification error on the validation set**.

Regularization

- Limiting the **maximum depth** of the tree is a form of **regularization**.
  - Likewise, the values for the other parameters, such as **min_impurity_decrease**, can be determined using a **validation set**.
- Another **regularization** technique for decision trees is known as **pruning**.
  - In a bottom-up fashion, **nodes are removed** if this reduces the **classification error on the validation set**.
- https://scikit-learn.org/dev/auto_examples/tree/plot_cost_complexity_pruning.html

Implementations

- **ID3** (Iterative Dichotomiser 3) **ID3**, **C4.5**, **C5.0** — by Ross Quinlan.
- **CART** (Classification And Regression Tree) — by Leo Breiman *et al.*
- **sklearn.tree.DecisionTreeClassifier**
    - `https://scikit-learn.org/stable/modules/tree.html`

```
from sklearn.tree import DecisionTreeClassifier

# ...

clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, y)
tree.plot_tree(clf)

# ...

clf.predict(X_new)
```

Random Forest

Random Forest

- **Ensemble** methods will be discussed with greater details later.

Random Forest

- **Ensemble** methods will be discussed with greater details later.
- “Although single decision trees can be excellent classifiers, increased accuracy often can be achieved by combining the results of a collection of decision trees.” [3]

Random Forest

- **Ensemble** methods will be discussed with greater details later.
- "Although single decision trees can be excellent classifiers, increased accuracy often can be achieved by combining the results of a collection of decision trees." [3]
- A **Random Forest** is a collection of decision trees.

Random Forest

- **Ensemble** methods will be discussed with greater details later.
- "Although single decision trees can be excellent classifiers, increased accuracy often can be achieved by combining the results of a collection of decision trees." [3]
- A **Random Forest** is a collection of decision trees.
    - Strategies to build a collection of trees:

Random Forest

- **Ensemble** methods will be discussed with greater details later.
- “Although single decision trees can be excellent classifiers, increased accuracy often can be achieved by combining the results of a collection of decision trees.” [3]
- A **Random Forest** is a collection of decision trees.
  - Strategies to build a collection of trees:
    - Creating new data sets using a **sampling with replacement procedure** (boostrap sampling);

Random Forest

- **Ensemble** methods will be discussed with greater details later.
- "Although single decision trees can be excellent classifiers, increased accuracy often can be achieved by combining the results of a collection of decision trees." [3]
- A **Random Forest** is a collection of decision trees.
  - Strategies to build a collection of trees:
    - Creating new data sets using a **sampling with replacement procedure** (boostrap sampling);
    - Using **a random subset** of the features for splitting (typically the square root of the total number of features);

Random Forest

- **Ensemble** methods will be discussed with greater details later.
- "Although single decision trees can be excellent classifiers, increased accuracy often can be achieved by combining the results of a collection of decision trees." [3]
- A **Random Forest** is a collection of decision trees.
  - Strategies to build a collection of trees:
    - Creating new data sets using a **sampling with replacement procedure** (boostrap sampling);
    - Using **a random subset** of the features for splitting (typically the square root of the total number of features);
    - Taking advantage of the **stochastic nature** of the procedure to build trees.

Random Forest

- **Ensemble** methods will be discussed with greater details later.
- “Although single decision trees can be excellent classifiers, increased accuracy often can be achieved by combining the results of a collection of decision trees.” [3]
- A **Random Forest** is a collection of decision trees.
    - Strategies to build a collection of trees:
        - Creating new data sets using a **sampling with replacement procedure** (boostrap sampling);
        - Using **a random subset** of the features for splitting (typically the square root of the total number of features);
        - Taking advantage of the **stochastic nature** of the procedure to build trees.
    - **Prediction**: the most common prediction (**majority vote**) amongst all the trees (the information can be used as an indication of the strength of the prediction).

Ensemble Learning

- Other ensemble learning techniques, such as **bagging**, **pasting**, **boosting**, and **stacking** will be discussed later.

Applications

Applications in bioinformatics

- “**Synthetic sick and lethal (SSL)** genetic interactions between genes A and B occur when the organism exhibits poor growth (or death) when both A and B are knocked out but not when either A or B is disabled individually.” [3]
- Determine the exon-intron structure of eukaryotic genes (gene finders). [3]
- In the study of gene expression profilling. [3]
- Cancer classification [4].

Prologue

Summary

- **Decision trees** can solve both tasks, **classification** and **regression**.

Summary

- **Decision trees** can solve both tasks, **classification** and **regression**.
- Handle a **mixture of data (feature) types**, real and categorical.

Summary

- **Decision trees** can solve both tasks, **classification** and **regression**.
- Handle a **mixture of data (feature) types**, real and categorical.
- **Interpretation** of the resulting model is high

Summary

- **Decision trees** can solve both tasks, **classification** and **regression**.
- Handle a **mixture of data (feature) types**, real and categorical.
- **Interpretation** of the resulting model is high
- **Stability** might be an issue, which can be alleviated when coupled with **ensemble** learning.

Next module

- **Hidden Markov Models**

References

- Gregor Stiglic, Simon Kocbek, Igor Pernek, and Peter Kokol.
  Comprehensive decision tree models in bioinformatics.
  PLoS One, 7(3):e33812, 2012.

- Pierre Geurts, Alexandre Irrthum, and Louis Wehenkel.
  Supervised learning with decision tree-based methods in computational and systems biology.
  Mol Biosyst, 5(12):1593–605, Dec 2009.

- Carl Kingsford and Steven L Salzberg.
  What are decision trees?
  Nat Biotechnol, 26(9):1011–3, Sep 2008.

- Dongsheng Che, Qi Liu, Khaled Rasheed, and Xiuping Tao.
  Decision tree and ensemble learning algorithms with their applications in bioinformatics.
  Adv Exp Med Biol, 696:191–9, 2011.

References

- Leander Schietgat, Celine Vens, Jan Struyf, Hendrik Blockeel, Dragi Kocev, and Saso Dzeroski.
  Predicting gene function using hierarchical multi-label decision tree ensembles.
  BMC Bioinformatics, 11:2, Jan 2010.
- Aurélien Géron.
  Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow.
  O'Reilly Media, 2nd edition, 2019.
- Andriy Burkov.
  The Hundred-Page Machine Learning Book.
  Andriy Burkov, 2019.

Marcel **Turcotte**
`Marcel.Turcotte@uOttawa.ca`

School of Electrical Engineering and **Computer Science** (EECS)
University of Ottawa