Some classifiers in Python

Dr. Jean Auriol

Postdoctoral Associate, University of Calgary

May 27, 2019

Dr. **Jean AURIOL**, Postdoctoral Associate, Department of Petroleum Engineering, University of Calgary.
Webpage: *http://cas.ensmp.fr/∼auriol/*

Curriculum

- **2015**: Civil engineer, **MINES ParisTech, PSL**, Paris, France
- **2015-2018**: PhD at **MINES ParisTech, PSL**, Paris, France and **University of Waterloo**,Canada.
  Robust design of backstepping controllers for systems of linear hyperbolic Partial Differential Equations.
- **2018**- Postdoctoral Associate at **University of Calgary**,Canada.
  Observation and control of subsurface processes during drilling

Expertise domains: Applied mathematics, control theory, hyperbolic PDEs, transport phenomena, drilling systems.

General outline

1. Introduction: difference between classification and regression
2. Nearest Neighbours classifier
3. Decision trees classifiers

General outline

1. Introduction: difference between classification and regression
2. Nearest Neighbours classifier
3. Decision trees classifiers

Regression: main ideas

Regression predictive modelling

Regression predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a continuous output variable (y).

Example: We have a **test bench** with the price and the size of different houses. Knowing the size of a new house, we want to know the corresponding price.

Regression: main ideas

Regression predictive modelling

Regression predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a continuous output variable (y).

Example: We have a **test bench** with the price and the size of different houses. Knowing the size of a new house, we want to know the corresponding price.

- A regression problem requires the prediction of a quantity.

Regression: main ideas

Regression predictive modelling

Regression predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a continuous output variable (y).

Example: We have a **test bench** with the price and the size of different houses. Knowing the size of a new house, we want to know the corresponding price.

- A regression problem requires the prediction of a quantity.
- Such a problem can have multiple input variables (**multivariate**).

Regression: main ideas

Regression predictive modelling

Regression predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a continuous output variable (y).

Example: We have a **test bench** with the price and the size of different houses. Knowing the size of a new house, we want to know the corresponding price.

- A regression problem requires the prediction of a quantity.
- Such a problem can have multiple input variables (**multivariate**).
- The **skill** of the model corresponds to the error in those predictions.

Example of scalar linear regression

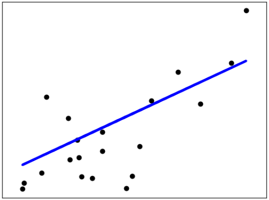
- Real system $y = f(x)$, f **unknown**.

Example of scalar linear regression

- Real system $y = f(x)$, f **unknown**.
- Set of n measurements $y_i = f(x_i)$.

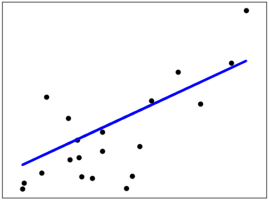
Example of scalar linear regression

- Real system $y = f(x)$, f **unknown**.
- Set of n measurements $y_i = f(x_i)$.
- Minimize the error between the estimation and the real value.



Example of scalar linear regression

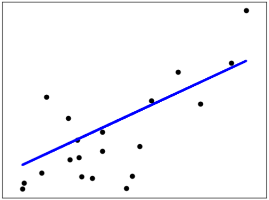
- Real system $y = f(x)$, f **unknown**.
- Set of n measurements $y_i = f(x_i)$.
- Minimize the error between the estimation and the real value.



Go from $y = f(x)$ **to the approximation** $y = ax$.

Example of scalar linear regression

- Real system $y = f(x)$, f **unknown**.
- Set of n measurements $y_i = f(x_i)$.
- Minimize the error between the estimation and the real value.



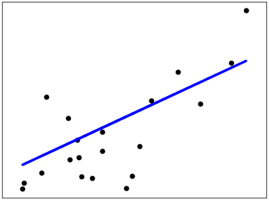
Go from $y = f(x)$ **to the approximation** $y = ax$.

Objective

Find a scalar a such that the approximation $y = ax$ is the best linear approximation of the real model in the sense of the $||\cdot||_2$-norm.

Example of scalar linear regression

- Real system $y = f(x)$, f **unknown**.
- Set of n measurements $y_i = f(x_i)$.
- Minimize the error between the estimation and the real value.



Go from $y = f(x)$ **to the approximation** $y = ax$.

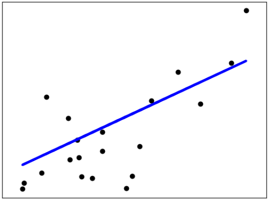
Objective

Find a scalar a such that the approximation $y = ax$ is the best linear approximation of the real model in the sense of the $||\cdot||_2$-norm.

- For a given a our estimator gives us the following estimations $\hat{y}_i = ax_i$.

Example of scalar linear regression

- Real system $y = f(x)$, f **unknown**.
- Set of n measurements $y_i = f(x_i)$.
- Minimize the error between the estimation and the real value.



Go from $y = f(x)$ **to the approximation** $y = ax$.

Objective

Find a scalar a such that the approximation $y = ax$ is the best linear approximation of the real model in the sense of the $||\cdot||_2$-norm.

- For a given a our estimator gives us the following estimations $\hat{y}_i = ax_i$.
- We want to find a that minimizes

$$\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 = \sum_{i=1}^{n}(ax_i - y_i)^2.$$

Classification: main ideas

Classification predictive modelling

Classification predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a discrete output variable (y) called **labels or categories**.

Example: An email of text can be classified as "spam" or "not spam".

Classification: main ideas

Classification predictive modelling

Classification predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a discrete output variable (y) called **labels or categories**.

Example: An email of text can be classified as "spam" or "not spam".

- A classification problem requires that examples be classified into one of two or more classes.

Classification: main ideas

Classification predictive modelling

Classification predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a discrete output variable (y) called **labels or categories**.

Example: An email of text can be classified as "spam" or "not spam".

- A classification problem requires that examples be classified into one of two or more classes.
- A problem with more than two classes is often called a **multi-class classification problem** .

Classification: main ideas

Classification predictive modelling

Classification predictive modelling is the task of **approximating** a mapping function (f) from input **explanatory** variables (X) to a discrete output variable (y) called **labels or categories**.

Example: An email of text can be classified as "spam" or "not spam".

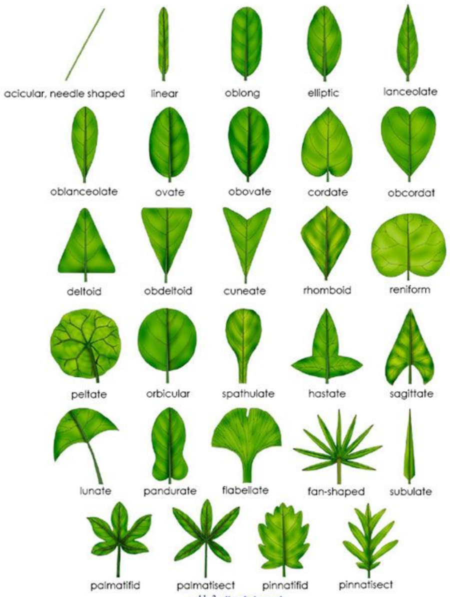
- A classification problem requires that examples be classified into one of two or more classes.
- A problem with more than two classes is often called a **multi-class classification problem** .
- The **classification accuracy** is the percentage of correctly classified examples out of all predictions made.

Examples of Classification Problems

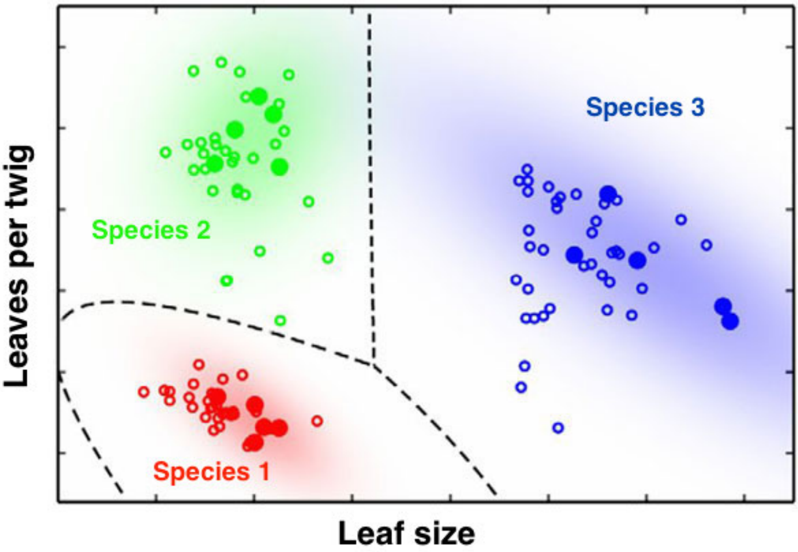
Problem	Features x	Class y	Used by?	Useful to have p(y|x)?
Spam email	Presence/absence of words in email	Spam or not	Google, Yahoo, Microsoft, etc	yes
Speech recognition	Acoustic/spectral features	Identity of word	IBM, Microsoft, Google, etc	yes
Loan Approval	Individuals' income, job, age, etc	Will default or not	Banks, financial companies	yes
Cancer screening	Image features at cell level	Cancerous or not?	Medical companies	yes
Personalized genomics	Gene expression data	Cancer or not	Bioinformatics startups	yes

UCIRVINE | UNIVERSITY of CALIFORNIA

Examples of classification problems



Examples of classification problems



Classification vs Regression

- **Classification** is the task of predicting a discrete class label.

Classification vs Regression

- **Classification** is the task of predicting a discrete class label.
- **Regression** is the task of predicting a continuous quantity.

Classification vs Regression

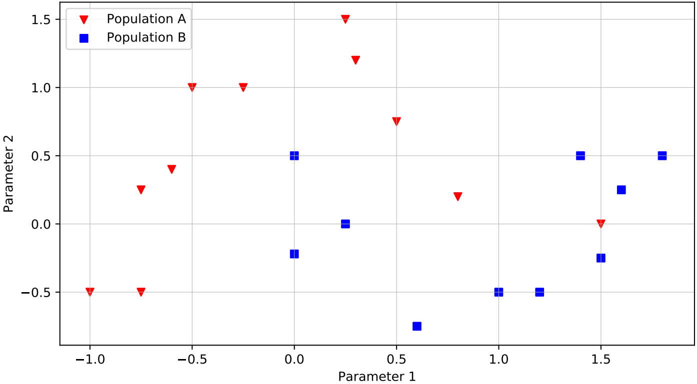
- **Classification** is the task of predicting a discrete class label.
- **Regression** is the task of predicting a continuous quantity.
- Some algorithms can be used for both classification and regression with small modifications (e.g. decision trees and artificial neural networks).

Classification vs Regression

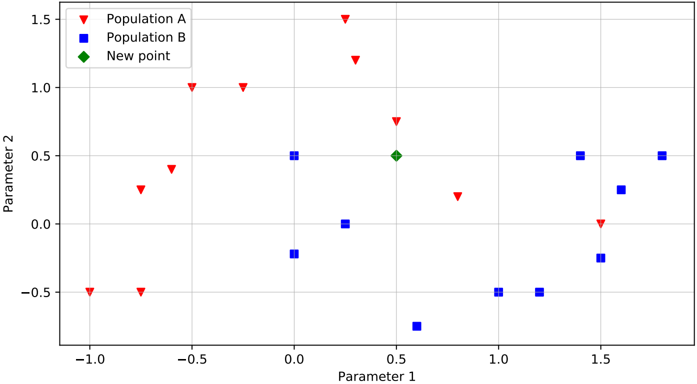
- **Classification** is the task of predicting a discrete class label.
- **Regression** is the task of predicting a continuous quantity.
- Some algorithms can be used for both classification and regression with small modifications (e.g. decision trees and artificial neural networks).
- Classification predictions can be evaluated using **accuracy**, whereas regression predictions can be evaluated using **mean squared error**.

General outline

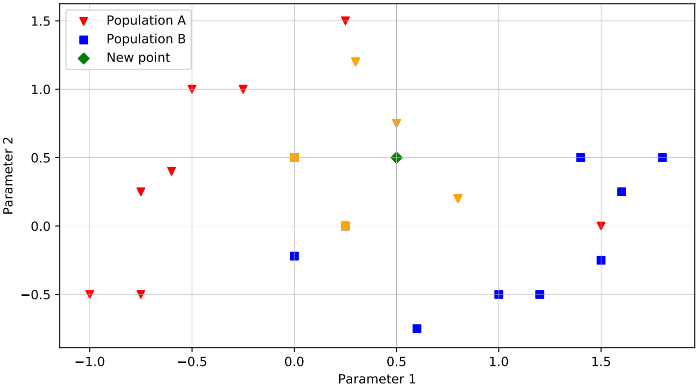
General idea of the algorithm



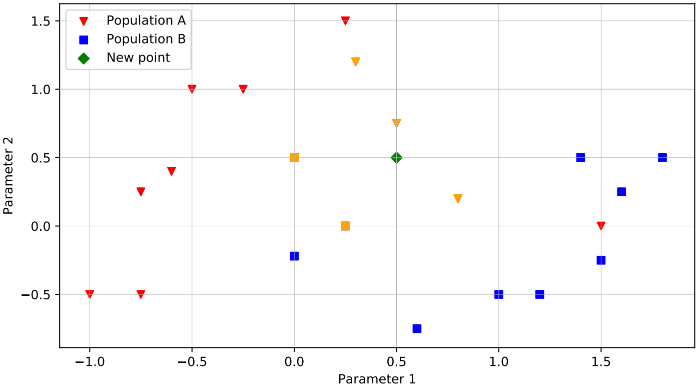
General idea of the algorithm



General idea of the algorithm



General idea of the algorithm

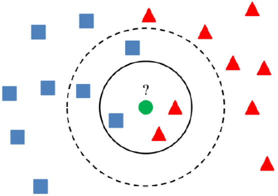


General idea

Find a **predefined number of training samples** closest in distance to the new point, and predict the label from these.

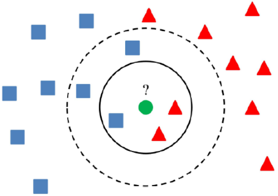
Description of the algorithm

- **Instance-based learning**: does not construct a model.



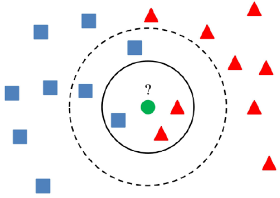
Description of the algorithm

- **Instance-based learning**: does not construct a model.
- Classification is computed from a **vote majority** .



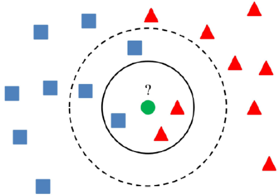
Description of the algorithm

- **Instance-based learning**: does not construct a model.
- Classification is computed from a **vote majority** .
- For each new point, we consider the k **nearest neighbours** and choose the majority class.



Description of the algorithm

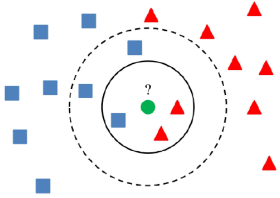
- **Instance-based learning**: does not construct a model.
- Classification is computed from a **vote majority** .
- For each new point, we consider the k **nearest neighbours** and choose the majority class.



- Simple algorithm, successfully used in a large number of problems.

Description of the algorithm

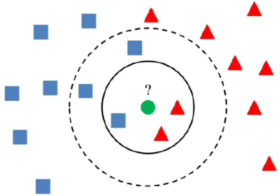
- **Instance-based learning**: does not construct a model.
- Classification is computed from a **vote majority** .
- For each new point, we consider the k **nearest neighbours** and choose the majority class.



- Simple algorithm, successfully used in a large number of problems.
- What does **near** mean?

Description of the algorithm

- **Instance-based learning**: does not construct a model.
- Classification is computed from a **vote majority** .
- For each new point, we consider the k **nearest neighbours** and choose the majority class.



- Simple algorithm, successfully used in a large number of problems.
- What does **near** mean?
- Algorithm that depends on k. What is a **good choice** of k?

Choice of the weights

- Euclidean metrics.

Uniform weights

The value assigned to a query point is computed from a simple majority vote of the nearest neighbours. **Each neighbour has the same weight**.

- Drawback occurs when the class distribution is skewed (domination of a class).

Choice of the weights

- Euclidean metrics.

Uniform weights

The value assigned to a query point is computed from a simple majority vote of the nearest neighbours. **Each neighbour has the same weight**.

- Drawback occurs when the class distribution is skewed (domination of a class).

Distance weights

We assign weights proportional to the inverse of the distance from the query point. **Nearer neighbours contribute more to the fit**.

Choice of the weights

- Euclidean metrics.

Uniform weights

The value assigned to a query point is computed from a simple majority vote of the nearest neighbours. **Each neighbour has the same weight**.

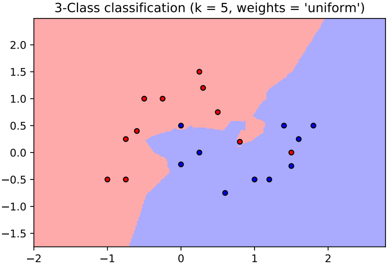
- Drawback occurs when the class distribution is skewed (domination of a class).

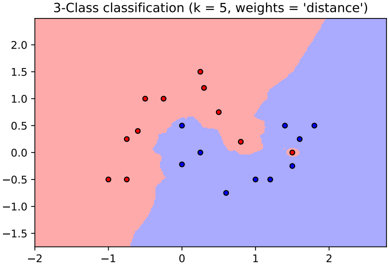
Distance weights

We assign weights proportional to the inverse of the distance from the query point. **Nearer neighbours contribute more to the fit**.

- **Not uniformly sampled data**: only consider the neighbours in a fixed radius r (pb in high dim).

- Classification domains.
- Algorithm neighbors.KNeighborsClassifier(k, weights)





How do we choose k?

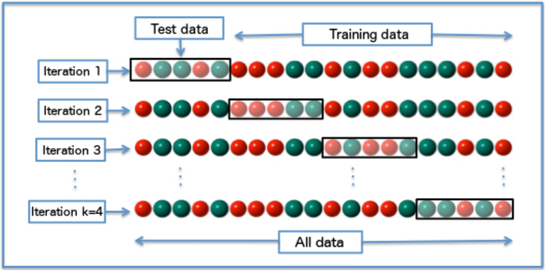
- The optimal choice of k is highly data-dependent.
- A larger k suppresses the effects of noise but makes the classification boundaries less distinct.

Cross-validation method

1. We partition our original data set into two subsets: the training set and the validation set .
2. The **training set** is used to define the considered neighbours.
3. We choose a value of k. We apply the algorithm on the validation set.
4. We compute the number of **misclassified** points.
5. We repeat, changing the value of k. We choose the optimal value of k.

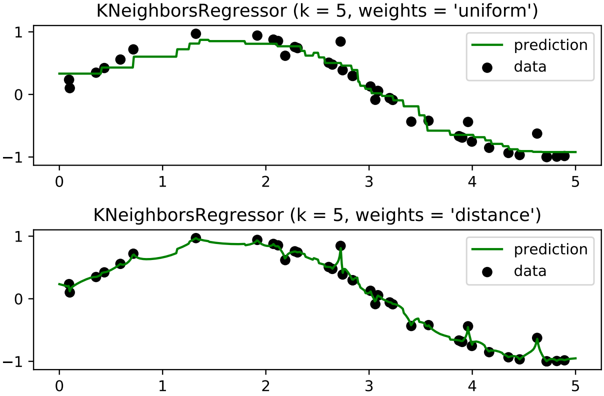
To reduce **variability**, multiple rounds of cross-validation are performed using different partitions. The results are averaged.

Cross-validation



Some words about the NN Regression

- Data labels are continuous (and not discrete).
- The label assigned to a query point is computed based on the mean of the labels of its nearest neighbours.



Implementation

- Different algorithms (brute-force, K-D trees, ball trees).
- Depends on the number of samples, query points, data structure.

Exercises 1-3.

General outline

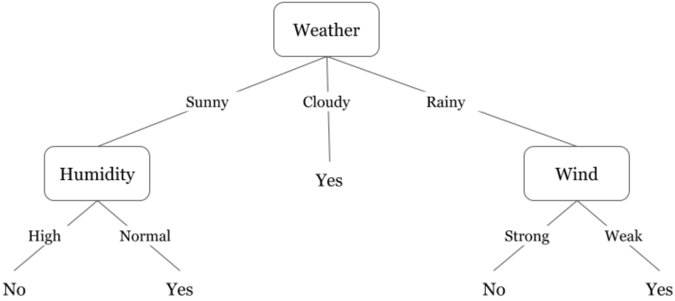
Example of a (simple) decision tree

- Should we play badminton?
- Data for the last 10 days.

Day	Weather	Temp	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	NO
2	Cloudy	Hot	High	Weak	YES
3	Sunny	Mild	Normal	Strong	YES
4	Cloudy	Mild	High	Strong	YES
5	Rainy	Mild	High	Strong	NO
6	Rainy	Cool	Normal	Strong	NO
7	Rainy	Mild	High	Weak	YES
8	Sunny	Hot	High	Strong	NO
9	Cloudy	Hot	Normal	Weak	YES
10	Rainy	Mild	High	Strong	NO

Example of a (simple) decision tree

- Should we play badminton?
- Data for the last 10 days.



Decision trees: general ideas

Definition

A **decision tree** is a tree where each node represents a feature (**attribute**), each link (**branch**) represents a decision (**rule**) and each **leaf** represents an outcome (categorical or continues value).

- **Classification tree:** the predicted outcome is the class to which the data belongs.
- **Regression tree:** the predicted outcome can be considered as a real number.

How to build a decision tree?

Which attribute do we need to pick first?

Determine the attribute that best classifies the training data; use this attribute at the root of the tree. Repeat this process for each branch.

How do we choose the best attribute?

Use the attribute with the highest **information gain.**

$\Rightarrow$ **Importance of metrics.**

- Measure of the **homogeneity** of the variable within the subsets.

Gini impurity

Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labelled if it was randomly labelled according to the distribution of labels in the subset.

Sum of the probability for each element to be chosen multiplied by the probability to be wrongly classified.

- We have m classes.
- p_i: fraction of items labelled with class i.
- $I_G = \sum_{i=1}^{m} p_i \sum_{k \neq i} p_k = 1 - \sum_{i=1}^{m} p_i^2$.

Day	Weather	Temp	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	NO
2	Cloudy	Hot	High	Weak	YES
3	Sunny	Mild	Normal	Strong	YES
4	Cloudy	Mild	High	Strong	YES
5	Rainy	Mild	High	Strong	NO
6	Rainy	Cool	Normal	Strong	NO
7	Rainy	Mild	High	Weak	YES
8	Sunny	Hot	High	Strong	NO
9	Cloudy	Hot	Normal	Weak	YES
10	Rainy	Mild	High	Strong	NO

- **Class weather:** $m = 3$ (Sunny, Cloudy, Rainy)

$$p_1 = 0.3, \;\; p_2 = 0.3, \;\; p_3 = 0.4$$

$$\Rightarrow \quad I_G = 1 - 0.3^2 - 0.3^2 - 0.4^2 = 0.66.$$

Day	Weather	Temp	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	NO
2	Cloudy	Hot	High	Weak	YES
3	Sunny	Mild	Normal	Strong	YES
4	Cloudy	Mild	High	Strong	YES
5	Rainy	Mild	High	Strong	NO
6	Rainy	Cool	Normal	Strong	NO
7	Rainy	Mild	High	Weak	YES
8	Sunny	Hot	High	Strong	NO
9	Cloudy	Hot	Normal	Weak	YES
10	Rainy	Mild	High	Strong	NO

- **Class Temperature:** $m = 3$ (Hot, Mild, Cool)

$$p_1 = 0.4, \ p_2 = 0.5, \ p_3 = 0.1$$

$$\Rightarrow \quad I_G = 1 - 0.4^2 - 0.5^2 - 0.1^2 = 0.58.$$

Metrics

Day	Weather	Temp	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	NO
2	Cloudy	Hot	High	Weak	YES
3	Sunny	Mild	Normal	Strong	YES
4	Cloudy	Mild	High	Strong	YES
5	Rainy	Mild	High	Strong	NO
6	Rainy	Cool	Normal	Strong	NO
7	Rainy	Mild	High	Weak	YES
8	Sunny	Hot	High	Strong	NO
9	Cloudy	Hot	Normal	Weak	YES
10	Rainy	Mild	High	Strong	NO

- **Class Humidity:** $m = 2$ (High, Normal)

$$p_1 = 0.7, \; p_2 = 0.3$$

$$\Rightarrow \quad I_G = 1 - 0.7^2 - 0.3^2 = 0.42.$$

Day	Weather	Temp	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	NO
2	Cloudy	Hot	High	Weak	YES
3	Sunny	Mild	Normal	Strong	YES
4	Cloudy	Mild	High	Strong	YES
5	Rainy	Mild	High	Strong	NO
6	Rainy	Cool	Normal	Strong	NO
7	Rainy	Mild	High	Weak	YES
8	Sunny	Hot	High	Strong	NO
9	Cloudy	Hot	Normal	Weak	YES
10	Rainy	Mild	High	Strong	NO

- **Class wind:** $m = 2$ (Weak, Strong)

$$p_1 = 0.4, \;\; p_2 = 0.6,$$
$$\Rightarrow \quad I_G = 1 - 0.4^2 - 0.6^2 = 0.48.$$

Metrics

Day	Weather	Temp	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	NO
2	Cloudy	Hot	High	Weak	YES
3	Sunny	Mild	Normal	Strong	YES
4	Cloudy	Mild	High	Strong	YES
5	Rainy	Mild	High	Strong	NO
6	Rainy	Cool	Normal	Strong	NO
7	Rainy	Mild	High	Weak	YES
8	Sunny	Hot	High	Strong	NO
9	Cloudy	Hot	Normal	Weak	YES
10	Rainy	Mild	High	Strong	NO

- The most relevant criterion is the weather.
- It splits that results in the purest daughter nodes.
- **Minimization** of the number of children nodes.

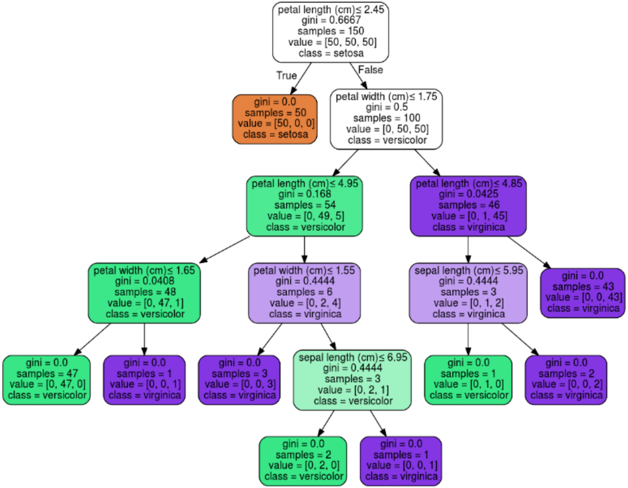
Entropy

Entropy is a characterization of the impurity of an arbitrary collection of examples.

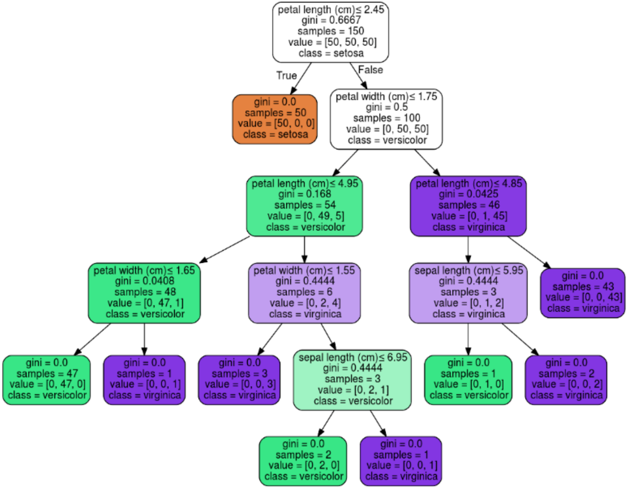
At each step we should choose the split that results in the purest daughter nodes.

$$I_G = -\sum_{i=1}^{m} p_i \log_2 p_i$$

For the last leaves, the Gini index/Entropy should be equal to zero.


petal length (cm)≤ 2.45
gini = 0.6667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm)≤ 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
petal length (cm)≤ 4.95
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
petal length (cm)≤ 4.85
gini = 0.0425
samples = 46
value = [0, 1, 45]
class = virginica
petal width (cm)≤ 1.65
gini = 0.0408
samples = 48
value = [0, 47, 1]
class = versicolor
petal width (cm)≤ 1.55
gini = 0.4444
samples = 6
value = [0, 2, 4]
class = virginica
sepal length (cm)≤ 5.95
gini = 0.4444
samples = 3
value = [0, 1, 2]
class = virginica
gini = 0.0
samples = 43
value = [0, 0, 43]
class = virginica
gini = 0.0
samples = 47
value = [0, 47, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica
gini = 0.0
samples = 3
value = [0, 0, 3]
class = virginica
sepal length (cm)≤ 6.95
gini = 0.4444
samples = 3
value = [0, 2, 1]
class = versicolor
gini = 0.0
samples = 1
value = [0, 1, 0]
class = versicolor
gini = 0.0
samples = 2
value = [0, 0, 2]
class = virginica
gini = 0.0
samples = 2
value = [0, 2, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica

Example of a complete tree (Iris dataset)



- Useless leaves...

Uses

Advantages

- Simple to understand and interpret.
- Requires little data preparation.
- Performs well with large datasets.
- Possible to validate a model using statistical tests.

Limitations

- Trees can be very non-robust.
- NP-complete problems.
- Can create over-complex trees.

Pruning

- A tree that is too large risks **overfitting** the training data.
- A small tree might not capture important structural information.

What is pruning?

Technique to remove nodes that do not provide additional information. **Pruning** should reduce the size of a learning tree without reducing predictive accuracy as measured by a cross-validation set.

There exist other techniques to improve the reliability of the prediction (**random forests, boosting** ...)

Exercise 4-5.