

# Orange

Origine, architecture

Interface

Accès aux données

Définir les traitements

Bibliothèque

Évaluation et comparaisons

Expérimentations

Performances

Exploration graphique

A.I. LAB – Université de Lubiana – Slovénie  
<http://magix.fri.uni-lj.si/orange/>

Culture « I.A. -- machine learning » -- ICML, JAIR

Code source libre en C++ (Licence GNU Scientific Library)

Site WEB avec de nombreux guides

Exécution stand-alone sous Windows

*Nécessite un interpréteur PYTHON*

*Les méthodes sont compilées sous forme de DLL*

# Orange

Origine, architecture

**Interface**

Accès aux données

Définir les traitements

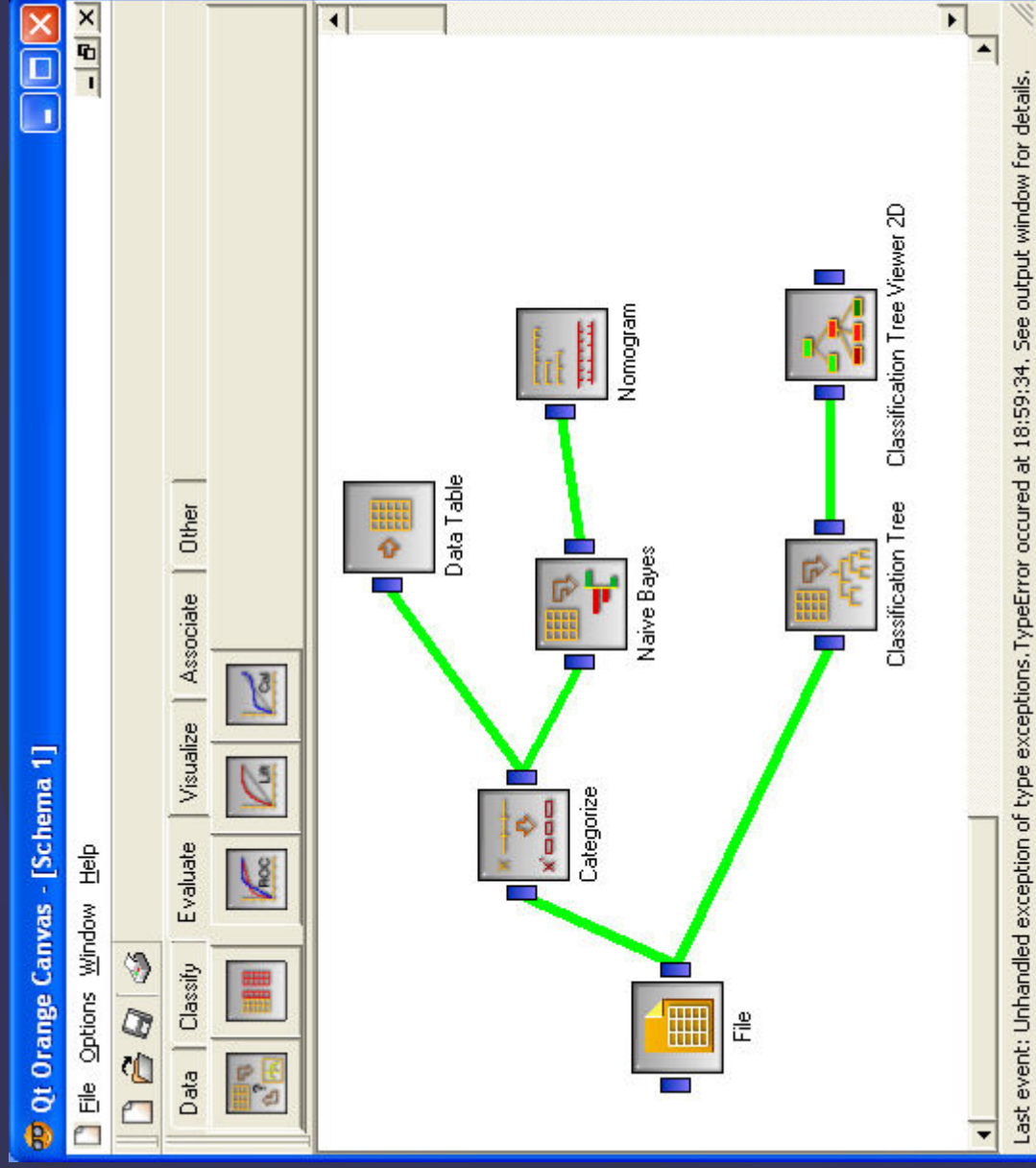
Bibliothèque

Évaluation et comparaisons

Expérimentations

Performances

Exploration graphique



« Simplifié mais beaucoup de non-dits »

# Orange

Origine, architecture

Interface

Accès aux données

Définir les traitements

Bibliothèque

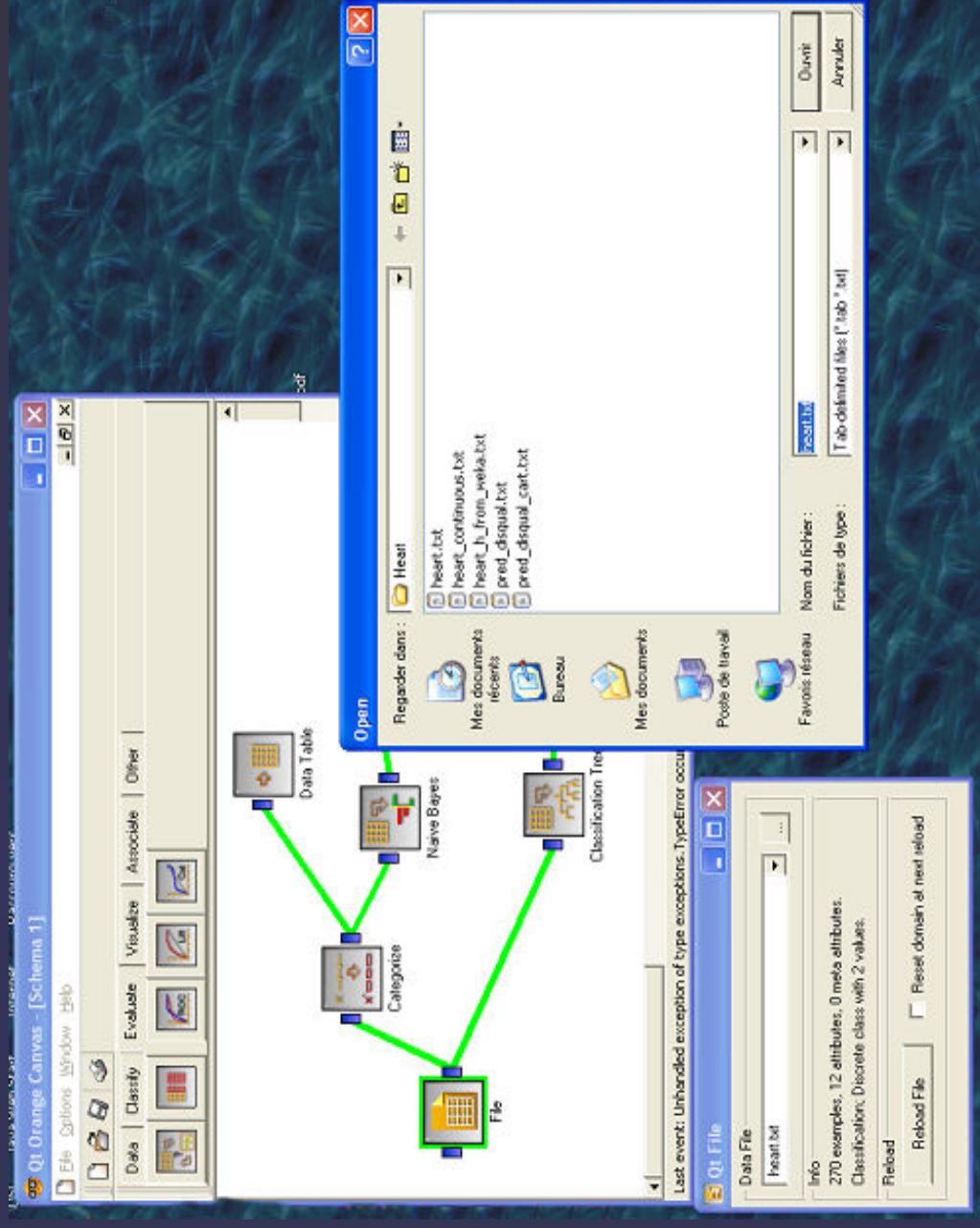
Évaluation et comparaisons

Expérimentations

Performances

Exploration graphique

Fichier texte, avec séparateur tabulation (export tableur par exemple)



Autres formats texte uniquement (C4.5, Assistant, ...)

# Orange

Origine, architecture

Interface

Accès aux données

**Définir les traitements**

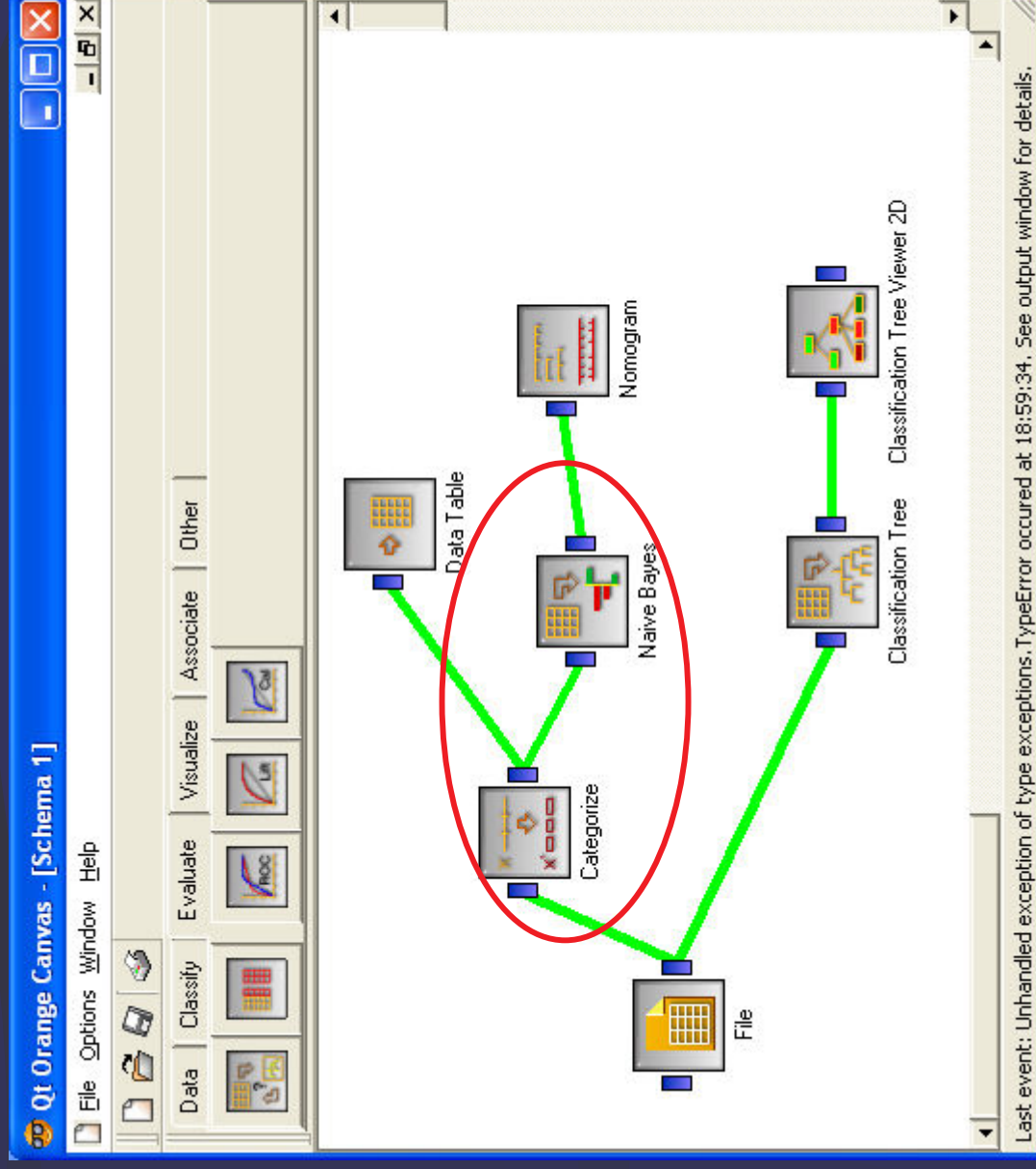
Bibliothèque

Évaluation et comparaisons

Expérimentations

Performances

Exploration graphique



- Plusieurs traitements en parallèle
- Combinaison de traitements

# Orange

Origine, architecture

Interface

Accès aux données

Définir les traitements

**Bibliothèque**

Évaluation et comparaisons

Expérimentations

Performances

Exploration graphique

- Description rapide des données et stat. descriptives 😊
- Traitement des données manquantes ?
- Restriction sur les individus (échantillon, condition) 😊
- Tests statistiques 😞
- Sélection des variables pour le supervisé (ranking) 😊
- Construction de variables (discrétisation, codage 0/1) 😊
- Apprentissage supervisé 😊
- Apprentissage non-supervisé 😊
- Règles d'association 😊
- Méthodes fact. (multidimensional scaling → matrice dist.) 😞
- Séries temporelles 😞



**Palette large et simplifiée**

# Orange

Origine, architecture

Interface

Accès aux données

Définir les traitements

Bibliothèque

Évaluation et comparaisons

Expérimentations

Performances

Exploration graphique

The screenshot displays the Orange3 workflow editor and the 'Qt Test Learners' dialog box. The workflow consists of the following components connected by green lines:

- File** (Data source)
- Categorize** (Data preprocessing)
- Data Table** (Data visualization)
- Naive Bayes** (Classifier)
- Classification Tree** (Classifier)
- Logistic Regression** (Classifier)
- Continue** (Workflow connector)
- Data Table (2)** (Data visualization)
- Test Learners** (Evaluation component)

The 'Qt Test Learners' dialog box is open, showing the 'Evaluation Results' table:

Classifier	CA	Sens	Spec	AUC
1 Classification Tree	0.7148	0.6167	0.7933	0.7097
2 Naive Bayes	0.8000	0.7583	0.8333	0.8706
3 Logistic regression	0.8296	0.8000	0.8533	0.8906

The dialog also includes settings for Sampling (Cross Validation, Number of Folds: 10, Leave-One-Out, Random Sampling, Repeat Train/Test: 10, Relative Training Set Size: 70%), Test on Train Data, and Test on Test Data. Statistics shown include Classification Accuracy, Sensitivity, Specificity, Area Under ROC Curve, Information Score, and Brier Score.

Possibilité de regrouper les résultats dans un seul composant  
Il reste la programmation sous PYTHON des traitements !!!

## EXPERIMENTATIONS : Exploiter les possibilités de python

Python est un langage de programmation interprété, multi-paradigme, ce qui signifie qu'il autorise la programmation impérative structurée, orientée objet, et fonctionnelle. Il est doté d'un typage dynamique (ce qui ne l'empêche pas de disposer d'un typage fort), d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions.  
([http://fr.wikipedia.org/wiki/Langage\\_de\\_programmation\\_Python](http://fr.wikipedia.org/wiki/Langage_de_programmation_Python))

```
# Description: Shows how to construct an orange.ClassifierFromExampleTable
# Category:   classification, lookup classifiers, constructive induction, feature construction
# Classes:   ClassifierByExampleTable, LookupLearner
# Uses:     monk1
# Referenced: lookup.htm

import orange

data = orange.ExampleTable("monk1")
a, b, e = data.domain["a"], data.domain["b"], data.domain["e"]

data_s = data.select([a, b, e, data.domain.classVar])
abe = orange.LookupLearner(data_s)

print len(data_s)
print len(abe.sortedExamples)

for i in abe.sortedExamples[:10]:
    print i
print

for i in abe.sortedExamples[:10]:
    print i, i.getClass().svalue
print

y2 = orange.EnumVariable("y2", values = ["0", "1"])
abe2 = orange.LookupLearner(y2, [a, b, e], data)
for i in abe2.sortedExamples[:10]:
    print i, i.getClass().svalue
print

y2 = orange.EnumVariable("y2", values = ["0", "1"])
abe2 = orange.LookupLearner(y2, [a, b], data)
for i in abe2.sortedExamples:
    print i, i.getClass().svalue
```

# Orange

Origine, architecture

Interface

Accès aux données

Définir les traitements

Bibliothèque

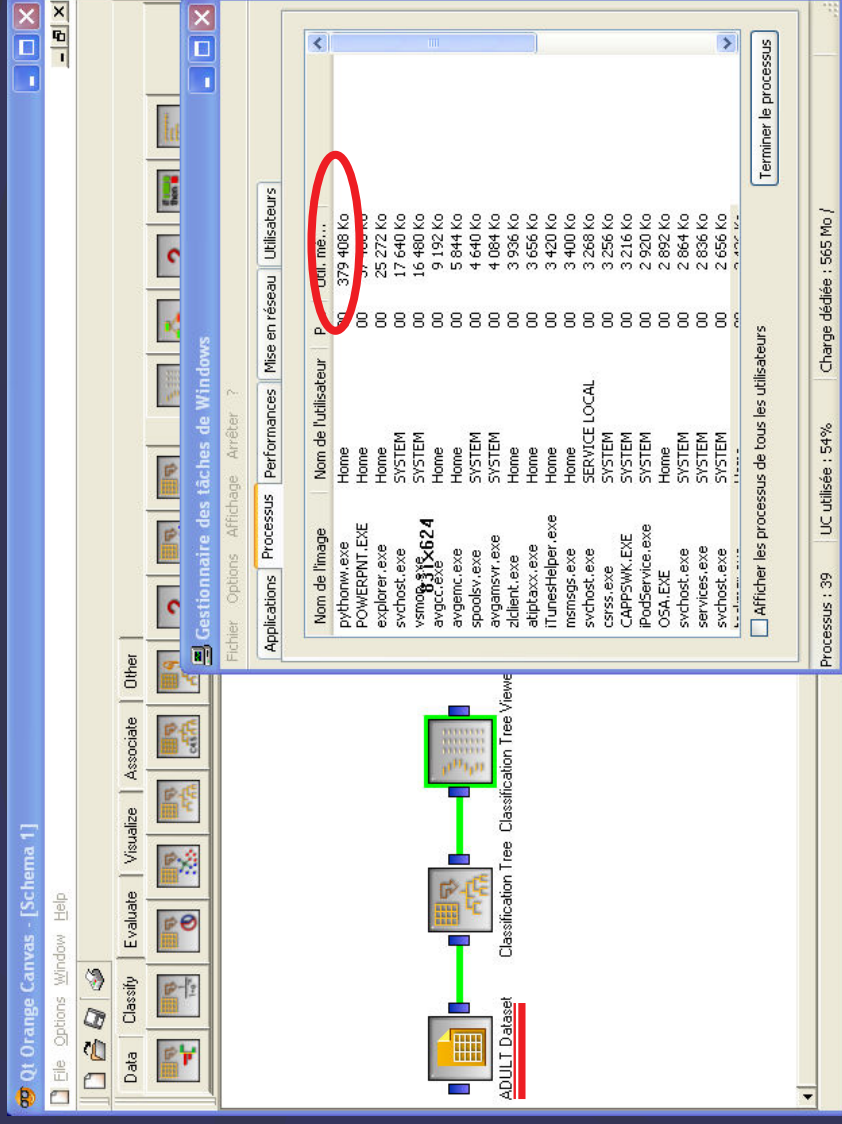
Évaluation et comparaisons

Expérimentations

**Performances**

Exploration graphique

- Pas de limitation théorique (espace adressable) 😊
  - Toutes les données en mémoire 😞
  - Méthodes = DLL compilées (C++) 😊
  - Lenteur et souci gestion mémoire sur grosses bases 😞
- (Python ? Passage des données entre composants ?...)





# Orange

Origine, architecture

Interface

Accès aux données

Définir les traitements

Bibliothèque

Évaluation et comparaisons

Expérimentations

Performances

Exploration graphique

Riche et INTERACTIF

