

TANAGRA

Spécifications, Développement et Promotion

Ricco RAKOTOMALALA
Université Lumière Lyon 2
Laboratoire ERIC

Ricco ?

Enseignant chercheur (CNU.27)

En poste à l'Université Lyon 2 – Faculté de Sciences Eco.

Recherche : Spécialisation Data Mining

- Arbres/Graphes d'induction
- Sélection de variables
- Autres aspects théoriques
- **Applications**

Développement et diffusion de logiciels libres

Diffusion de supports de cours

PLAN

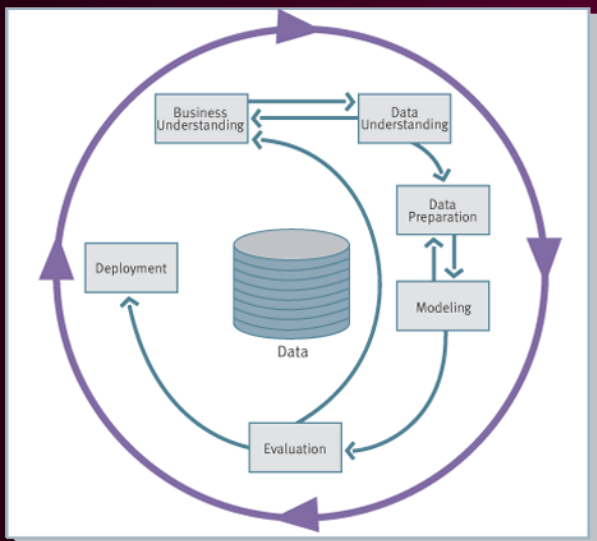
1. Data Mining
2. Logiciel libre pour quel public
3. Avant TANAGRA
4. TANAGRA
 - a. Spécifications
 - b. Développement
 - c. Promotion
5. Quelques scénarios de traitements
6. Et les autres logiciels libres ?
7. A posteriori...

1. Data Mining

CRISP-DM 1.0, Step-by-step Data Mining Guide, SPSS Publication

Le processus ECD : Extraction de connaissances à partir de données

KDD - Knowledge Discovery in Databases

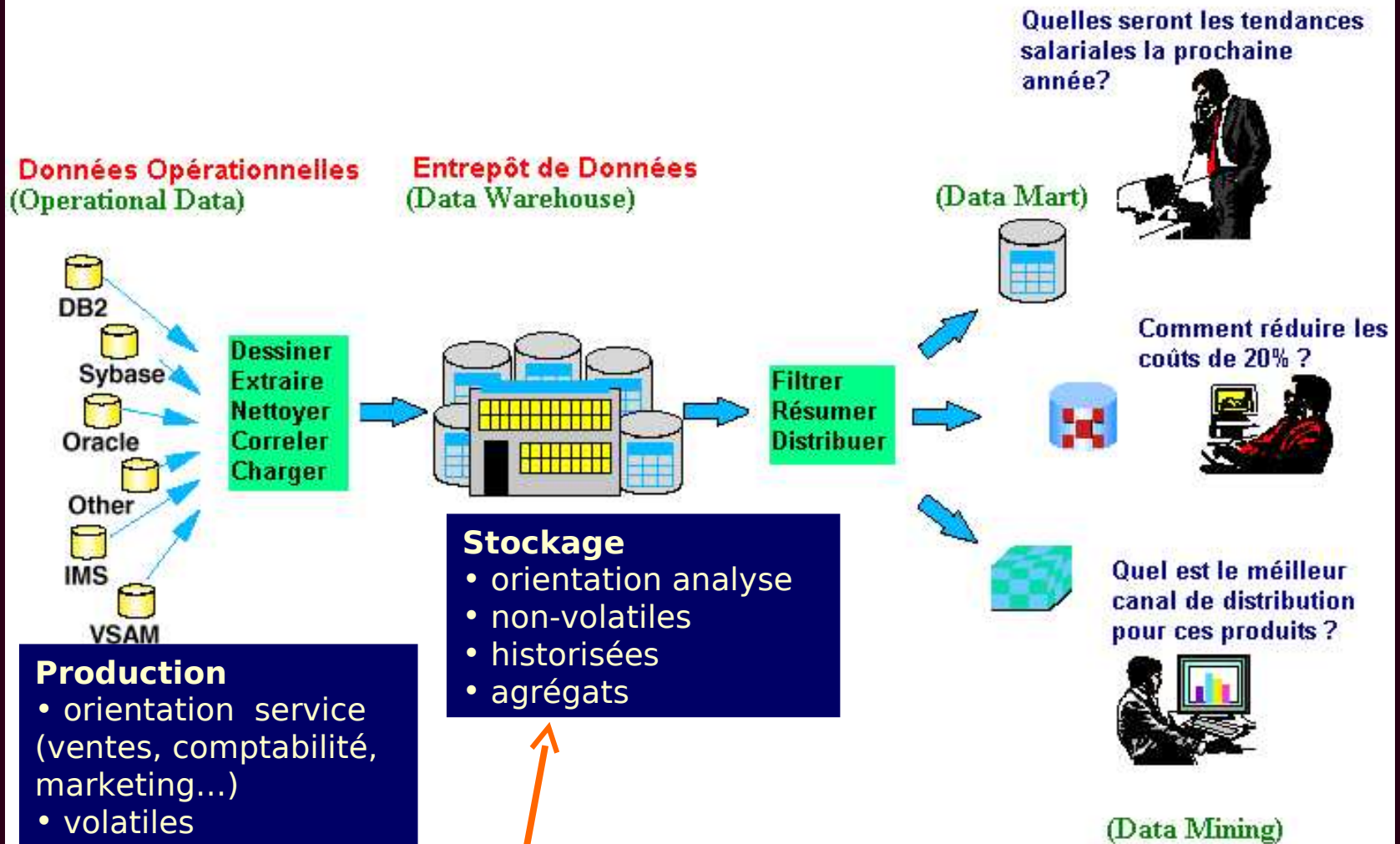


	Data Understanding	Data Preparation	Modeling	Evaluation	Deployment
Initial Data Collection	Select Data <i>Rationale for Inclusion/Exclusion</i>	Clean Data <i>Data Cleaning Report</i>	Select Modeling Techniques <i>Modeling Technique Modeling Assumptions</i>	Evaluate Results <i>Assessment of Data Mining Results w.r.t. Business Success Criteria Approved Models</i>	Plan Deployment <i>Deployment Plan</i>
Data Description	Construct Data <i>Derived Attributes Generated Records</i>	Integrate Data <i>Merged Data</i>	Generate Test Design <i>Test Design</i>	Review Process <i>Review of Process</i>	Plan Monitoring and Maintenance <i>Monitoring and Maintenance Plan</i>
Assess Situation <i>Inventory of Resources Requirements, Assumptions, and Constraints Risks and Contingencies Terminology Costs and Benefits</i>	Explore Data <i>Data Exploration Report</i>	Format Data <i>Reformatted Data Dataset Dataset Description</i>	Build Model <i>Parameter Settings Models Model Descriptions</i>	Determine Next Steps <i>List of Possible Actions Decision</i>	Produce Final Report <i>Final Report Final Presentation</i>
Determine Data Mining Goals <i>Data Mining Goals Data Mining Success Criteria</i>	Verify Data Quality <i>Data Quality Report</i>		Assess Model <i>Model Assessment Revised Parameter Settings</i>		Review Project <i>Experience Documentation</i>
Produce Project Plan <i>Project Plan Initial Assessment of Tools and Techniques</i>					

Quelles spécificités ?

Sources de données : Les bases de données de l'entreprise

Construire une Infrastructure d'Information Intelligente pour l'Entreprise



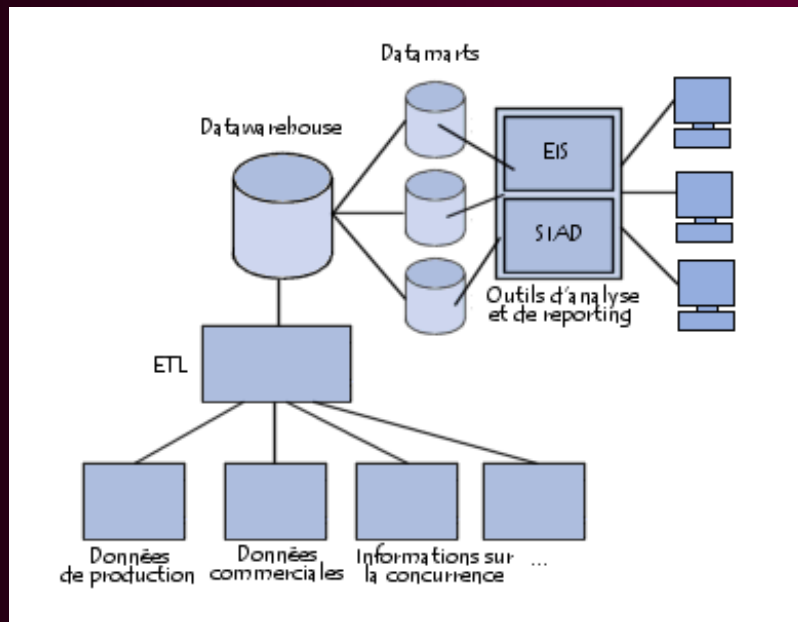
La gestion de la volumétrie devient un aspect important !

Data Mining et Informatique Décisionnelle

Business Intelligence

L'**informatique décisionnelle** (BI) désigne les moyens, les outils et les méthodes qui permettent de collecter, consolider, modéliser et restituer les données d'une entreprise en vue d'offrir une aide à la décision et de permettre aux responsables de la stratégie d'une entreprise d'avoir une vue d'ensemble de l'activité traitée.

(http://fr.wikipedia.org/wiki/Informatique_décisionnelle)



- Sélectionner les données (par rapport à un sujet et/ou une période)
- Trier, regrouper ou répartir ces données selon certains critères
- Élaborer des calculs récapitulatifs « simples » (totaux, moyennes conditionnelles, etc.)
- Présenter les résultats de manière synthétique (graphique et/ou tableaux de bord) → REPORTING

La notion de modélisation « statistique » (apprentissage, exploration de données) est mise de côté → Data Mining

Quelles spécificités ?

Les techniques, selon leur origine

Statistiques

Théorie de l'estimation, tests
Économétrie

Maximum de vraisemblance et moindres carrés
Régression logistique, ...

Analyse de données
(Statistique exploratoire)
Description factorielle
Discrimination
Clustering

Méthodes géométriques, probabilités
ACP, ACM, Analyse discriminante, CAH, ...

	var 1	var 2	...	var J
individu 1				
individu 2		valeurs		
...				
individu n				

Informatique

« Machine Learning »
Apprentissage symbolique
Reconnaissance de formes

Une étape de l'intelligence artificielle
Réseaux de neurones, algorithmes génétiques...

Informatique

(Base de données)
Exploration des bases de données

Volumétrie
Règles d'association, motifs fréquents, ...

Très souvent, ces méthodes reviennent à optimiser les mêmes critères, mais avec des approches / formulations différentes

Quelles spécificités ?

Les techniques selon leurs finalités

Description :

Trouver un résumé des données qui soit plus intelligible

- statistique descriptive
- analyse factorielle

Ex : moyenne d'âge des personnes présentant un cancer du sein

Structuration :

Faire ressurgir des groupes « naturels » qui représentent des entités particulières

- **classification** (clustering, apprentissage non-supervisé)

Ex : découvrir une typologie de comportement des clients d'un magasin

Méthodes de Data Mining

Explication :

Prédire les valeurs d'un attribut (endogène) à partir d'autres attributs (exogènes)

- régression
- **apprentissage supervisé**

Ex : prédire la qualité d'un client (rembourse ou non son crédit) en fonction de ses caractéristiques (revenus, statut marital, nombre d'enfants, etc.)

Association :

Trouver les ensembles qui reviennent souvent ensemble

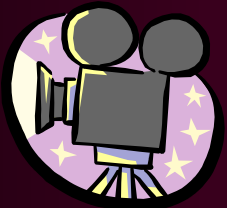
- **règles d'association**
- **motifs fréquents**

Ex : rayonnage de magasins, les personnes qui achètent du poivre achètent également du sel

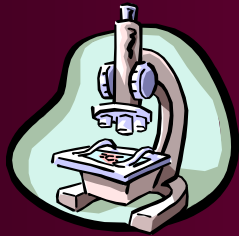
Les méthodes sont le plus souvent complémentaires !

Quelles spécificités ?

Traitement des données non structurées



Rôle fondamental de la préparation des données !



	var 1	var 2	...	var J
individu 1				
individu 2		valeurs		
...				
individu n				



Prédiction
Structuration
Description
Association

Les applications

Filtrage automatique des e-mails (spams, ...)

Reconnaissance de la langue à une centrale téléphonique

Analyse des mammographies

Etc.

Quel espace pour les logiciels libres ?

Aspects du Data Mining prolifiques en développement

Développer des méthodes au cœur des entrepôts de données

Les B.D. sont surtout intéressés par le développement des plate-formes B.I.

Proximité très (trop) forte avec les applications industrielles (ORACLE, SQL-Server...)

Développement lourds, peu valorisables pour l' « apprentissage automatique »

Traitement des données non structurées

Trop spécifique – Impossible de développer un outil générique

Proximité des applications industrielles

Développer des outils génériques de traitement de données

Intégrer des méthodes avec des finalités (origines) différentes

Pouvoir les faire coopérer entre elles

Tester et diffuser une nouvelle méthode publiée

Développement de la plate-forme peu onéreuse, ce qui est difficile c'est le développement des algorithmes de traitement (ex. YALE, KNIME et ALPHA

MINER reposent en partie sur le moteur WEKA)



2. Quel public pour le logiciel libre ?

Un logiciel pour l'enseignement : le profil « chargé d'études »

Les cours, explication des méthodes, **outil pédagogique**

Les études « réelles » - les « dossiers » - les chercheurs des autres domaines

(cf. tutoriaux études de cas)



Une plate-forme pour la recherche

Plate-forme d'expérimentation – Tester et comparer des méthodes

Modularité et accès au code – Programmer ses propres méthodes

(cf. tutoriaux évaluation des méthodes)

Un outil pédagogique pour l'apprentissage de la programmation

Spécifications et conception de ce type de logiciel - Apprendre par l'exemple

Connaître les outils et les bibliothèques types

Sujets de stages pour les étudiants

*(cf. le projet **Tanagra+**)*

Valider le code = valider les publications

Comparer les résultats

Lecture du code par d'autres chercheurs (ex. du text mining par SD)

Reproduire « exactement » les expérimentations (ex. tirage aléatoire)



Comparer les implémentations

Comparer les interprétations d'un même problème (ex. Relief WEKA)

Optimiser le code avec différentes versions

Outil ouvert = Outil vivant

Introduire ses propres algorithmes

Discuter sur la base de prototypes et d'évolutions

Monter et partager des bibliothèques types (ex. générateurs aléatoires, fonctions de répartition, pourquoi pas des bibliothèques de DATA MINING ?...)

3. Avant TANAGRA

SIPINA – Une longue lignée de logiciels dédiés aux Graphes d'Induction

Idée initiale : implémenter les Graphes d'Induction (Graphes Latticiels)

Picard (1965), Terrenoire, Tounissoux,..., Zighed (1985)



Zighed (1985...)

Piloté par interpréteur de commandes

Format spécifique de fichiers

Méthode SIPINA

Pas de diffusion institutionnelle



Version 1.0 à 2.5 – 16 bits (W3.0 et +)

Zighed (1994-1997)

Ponsard, Bac (1994-1995) – Rakotomalala (1996 – 1997)

Piloté par menu

Format spécifique de fichiers

Généralisation aux arbres de décision

Diffusée sur Internet



Version « Recherche » (ou version 3.0) – 32 bits (W95 et +)

Rakotomalala (1998 - 2000)...

Piloté par menu

Gestion « performante » des données, accès à différents formats

Généralisation aux autres méthodes supervisées (RNA, LDA, Règles, etc.)

Diffusée sur Internet



Version 4.0 (MCubiX)

Société Diagnos (2001...)

Diagramme de traitements (généralisation de la « filière »)

Interfacée avec une SGBD (Interbase)

Généralisation à toutes les techniques statistiques (méthodes factorielles, clustering, etc.)

Diffusion commerciale

4.a. TANAGRA - Spécifications

TANAGRA (2003) – Nouveau départ ou recommencement ?

Définir un cahier de charges le plus précis possible



Miser sur la simplicité de fonctionnement

1. Installation simplifiée – Pas de serveurs lourds à installer
2. Gestion simplifiée des données -- Format texte et accès au format tableur
3. Fonctionnement par **diagramme de traitements**
4. Couvrir les statistiques, l'analyse de données et le data mining. De manière unifiée.
5. Résultats lisibles, en adéquation avec les « standards », possibilité de les reprendre dans un traitement de texte ou un tableur par copier/coller

Mettre définitivement de côté les aspects « professionnels » des logiciels de D.M.

1. Interfaçage fort avec les SGBD
2. Déploiement et mise en production des résultats
3. Reporting dynamique et performant
4. Exploration graphique évoluée et interactive des données

Simplicité également pour le programmeur

1. Simplifier à l'extrême le code permettant d'ajouter une nouvelle méthode d'analyse
2. Minimiser le code dédié à la gestion des données et de l'interface
3. Pouvoir intégrer facilement n'importe quelle technique traitant des tableaux « individus x variables »

Simplifier les interfaces

Le diagramme de traitements

The screenshot displays the Sipina Research Version software interface. The main window shows a decision tree diagram for classifying Iris species based on petal length and petal width. The tree structure is as follows:

- Root node: **pet_length** with a split at ≥ 2.45 .
 - Left branch (< 2.45): Leaf node with 50 (33%) blue, 50 (33%) green, and 50 (33%) red. Legend: Iris-setosa (blue), Iris-versicolor (green), Iris-virginica (red).
 - Right branch (≥ 2.45): Internal node **pet_width** with a split at ≥ 1.75 .
 - Left branch (< 1.75): Leaf node with 0 (00%) blue, 49 (91%) green, and 5 (09%) red.
 - Right branch (≥ 1.75): Leaf node with 0 (00%) blue, 1 (02%) green, and 45 (98%) red.

The interface also includes a menu bar (Induction method, Analysis, Tree management, View, Window, Help), a toolbar, and a data table with columns: sep_length, sep_width, pet_length, pet_width, type. The status bar at the bottom indicates 'Improved ChAID (Tschuprow Goodness of Split)' and 'Time : 297'.

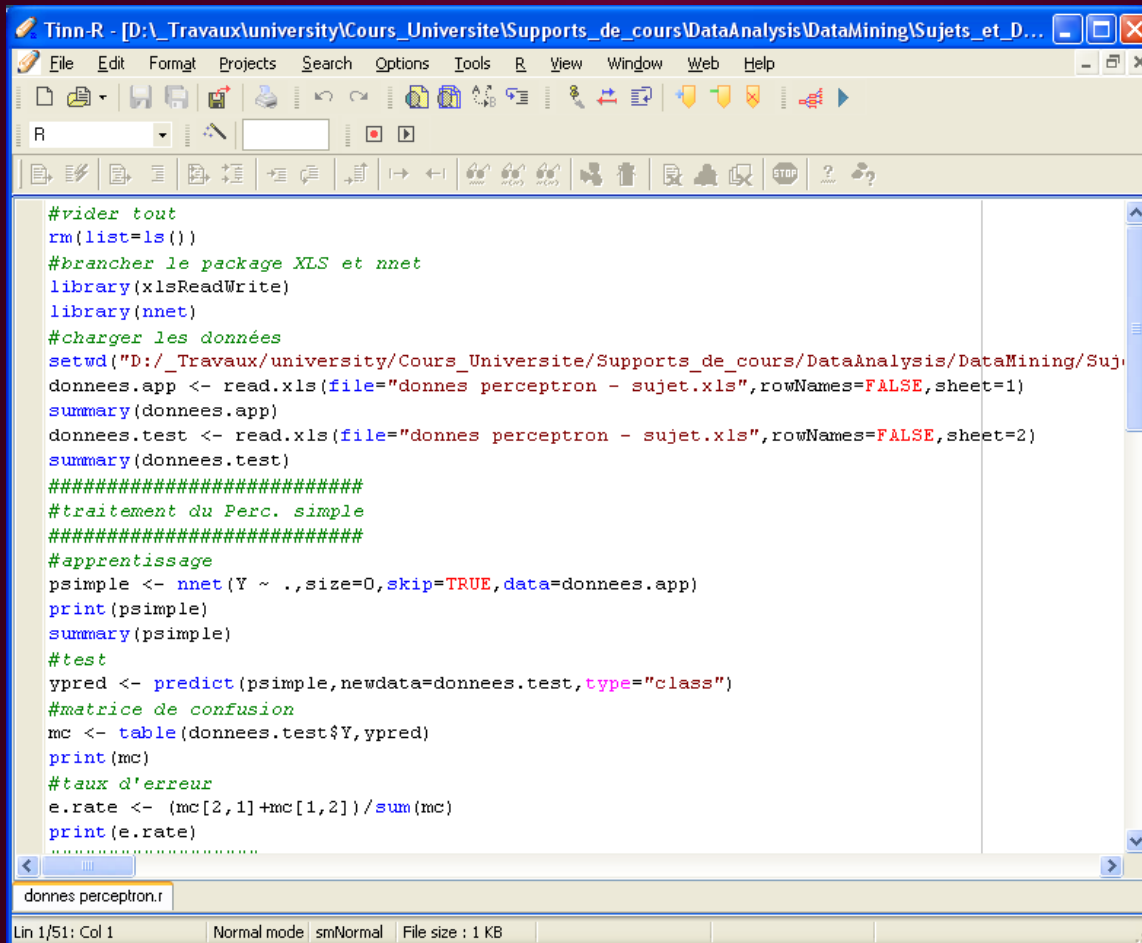
Pilotage par menu

Simple au premier abord mais ingérable dès que le logiciel gagne en complexité

Impossibilité de garder la trace d'une analyse complète et donc de la reproduire

Exige une documentation complète et constamment à jour

(Open Stat & Stat 4U sont dans la même situation)



```
#vider tout
rm(list=ls())
#brancher le package XLS et nnet
library(xlsReadWrite)
library(nnet)
#charger les données
setwd("D:/_Travaux/university/Cours_Universite/Supports_de_cours/DataAnalysis/DataMining/Sujets_et_D...")
donnees.app <- read.xls(file="donnees perceptron - sujet.xls",rowNames=FALSE,sheet=1)
summary(donnees.app)
donnees.test <- read.xls(file="donnees perceptron - sujet.xls",rowNames=FALSE,sheet=2)
summary(donnees.test)
#####
#traitement du Perc. simple
#####
#apprentissage
psimple <- nnet(Y ~ .,size=0,skip=TRUE,data=donnees.app)
print(psimple)
summary(psimple)
#test
ypred <- predict(psimple,newdata=donnees.test,type="class")
#matrice de confusion
mc <- table(donnees.test$Y,ypred)
print(mc)
#taux d'erreur
e.rate <- (mc[2,1]+mc[1,2])/sum(mc)
print(e.rate)
.....
```

donnees perceptron.r

Lin 1/51: Col 1 Normal mode smNormal File size : 1 KB

Langage de programmation

Toute la puissance d'un langage de programmation

L'accès au langage est une barrière à l'entrée qui rebute certains

L'intégration dans R est certainement la meilleure solution dans ce cas

TANAGRA 1.4.20 - [A priori 1]

File Diagram Component Window Help

Default title

- Dataset (banque.txt)
 - Define status 1
 - A priori 1

RULES

Number of rules : 8

N°	Antecedent	Consequent	Lift	Support	Confidence
1	"habit=locataire" - "csp=cadre_moyen" - "port_action=oui"	"accord=oui"	1.192	0.343	0.861
2	"habit=locataire" - "port_action=oui"	"accord=oui"	1.191	0.404	0.860
3	"Age=ancien"	"habit=locataire" - "csp=cadre_moyen"	1.161	0.348	0.932
4	"port_action=oui"	"accord=oui"	1.152	0.449	0.832
5	"csp=cadre_moyen" - "port_action=oui"	"accord=oui"	1.151	0.374	0.831
6	"csp=cadre_moyen" - "port_action=oui"	"habit=locataire" - "accord=oui"	1.146	0.343	0.764
7	"Age=ancien"	"csp=cadre_moyen"	1.111	0.359	0.959
8	"habit=locataire" - "Age=ancien"	"csp=cadre_moyen"	1.110	0.348	0.958

Components

Data visualization	Statistics	Nonparametric statistics	Instance selection	Feature construction
Feature selection	Regression	Factorial analysis	PLS	Clustering
Spv learning	Meta-spv learning	Spv learning assessment	Scoring	Association

A priori
 A priori PT
 Spv Assoc Rule
 A priori MR
 Assoc Outlier
 Spv Assoc Tree

Diagramme de traitements

« Programmation » visuelle - Enchaîner les traitements

Mise à jour facilitée du logiciel par adjonction de composants

Garder la trace d'une analyse complète et pouvoir la reproduire facilement

Possibilité de fragmenter la documentation par composants

C'est un *standard* (ex. SPAD, SAS-EM, SPSS-CLEM... WEKA, ORANGE, ...)

Interface et fonctionnalités

Fenêtres interactives et personnalisées vs. sorties textes généralisées

The screenshot displays the Sipina Research software interface. The main window shows a decision tree with nodes for 'pet_length' and 'pet_width'. A pop-up window titled 'Informations on : Level 2, Node 2' is open, showing a table of 'Descriptors' importance and a 'Split suggestion' table.

Informations on : Level 2, Node 2
IF pet_length >= 2.45

Characterization Descriptors' importance

Select an attribute to view the suggested split
Double-click to split with the selected attribute

	Goodness of split	Correlation	Accept or Reject
pet_width	0.55111060	0.5511	Accept
pet_length	0.52825876	0.5283	Accept
sep_length	0.15113515	0.1511	Reject
sep_width	0.05026315	0.0503	Reject

Split suggestion

	< 1.75	>= 1.75
Iris-setosa	0	0
Iris-versicol	49	1
Iris-virginica	5	45

100 examples (66.67% of the learning set)

Fenêtres personnalisées pour chaque traitement

Très user-friendly

Mais programmation (fastidieuse) qui éloigne du développement des méthodes

1 méthode = 1 fenêtre de visualisation nouvelle à développer

Optimisation très contraignante, occupation mémoire, etc.

TANAGRA 1.4.20 - [Supervised Learning 1 (C-RT)]

File Diagram Component Window Help

Analysis

- Dataset (tan8F.txt)
 - Define status 1
 - Supervised Learning 1 (C-RT)

Trees sequence (# 4)

N°	# Leaves	Err (growing set)	Err (pruning set)
4	1	0.6400	0.7200
3	2	0.3000	0.4000
2	3	0.0300	0.0600
1	5	0.0200	0.0600

Tree description

Number of nodes	5
Number of leaves	3

Decision tree

- pet_length < 2.4500 then type = **Iris-setosa** (100.00 % of 34 examples)
- pet_length >= 2.4500
 - pet_width < 1.7500 then type = **Iris-versicolor** (92.31 % of 39 examples)
 - pet_width >= 1.7500 then type = **Iris-virginica** (100.00 % of 27 examples)

Components

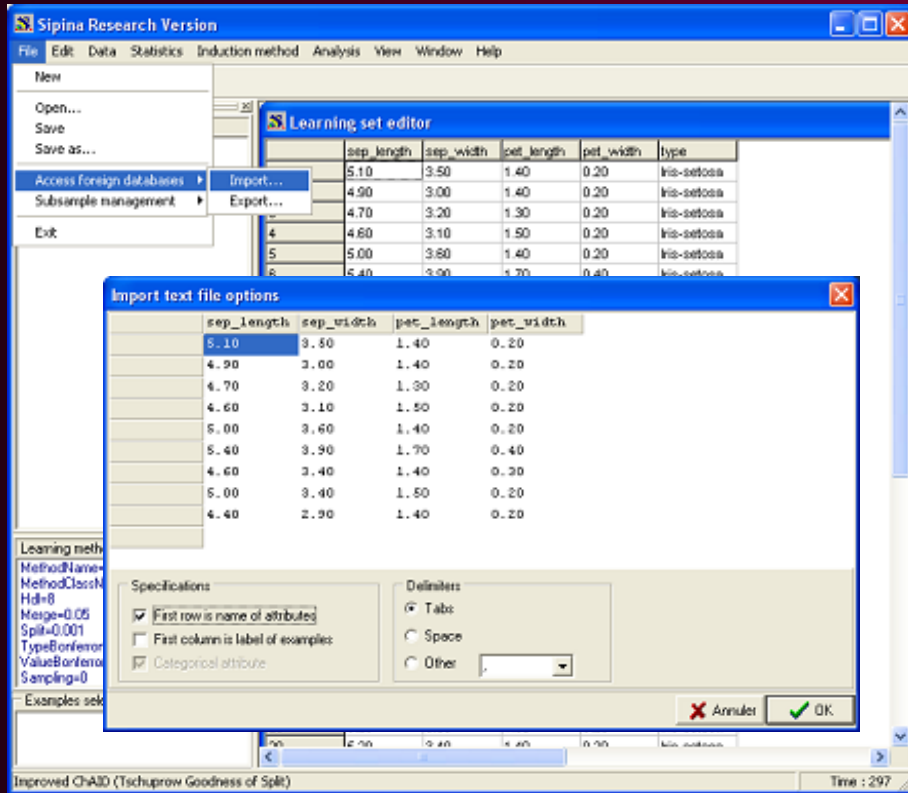
Data visualization	Statistics	Nonparametric statistics	Instance selection
Feature construction	Feature selection	Regression	Factorial analysis
PLS	Clustering	Spv learning	Meta-spv learning
Spv learning assessment	Scoring	Association	

Binary logistic regression
 C4.5
 C-PLS
 C-RT

Fenêtre standardisée – Format texte (agrémenté de HTML)
 Rébarbatif (?) mais conforme aux descriptions des méthodes dans les ouvrages
 Occupation mémoire quasi-nulle
 Copier coller standardisé vers les tableurs et traitement de texte
 Standardisation de la programmation des méthodes

Interface et fonctionnalités

Accès aux données



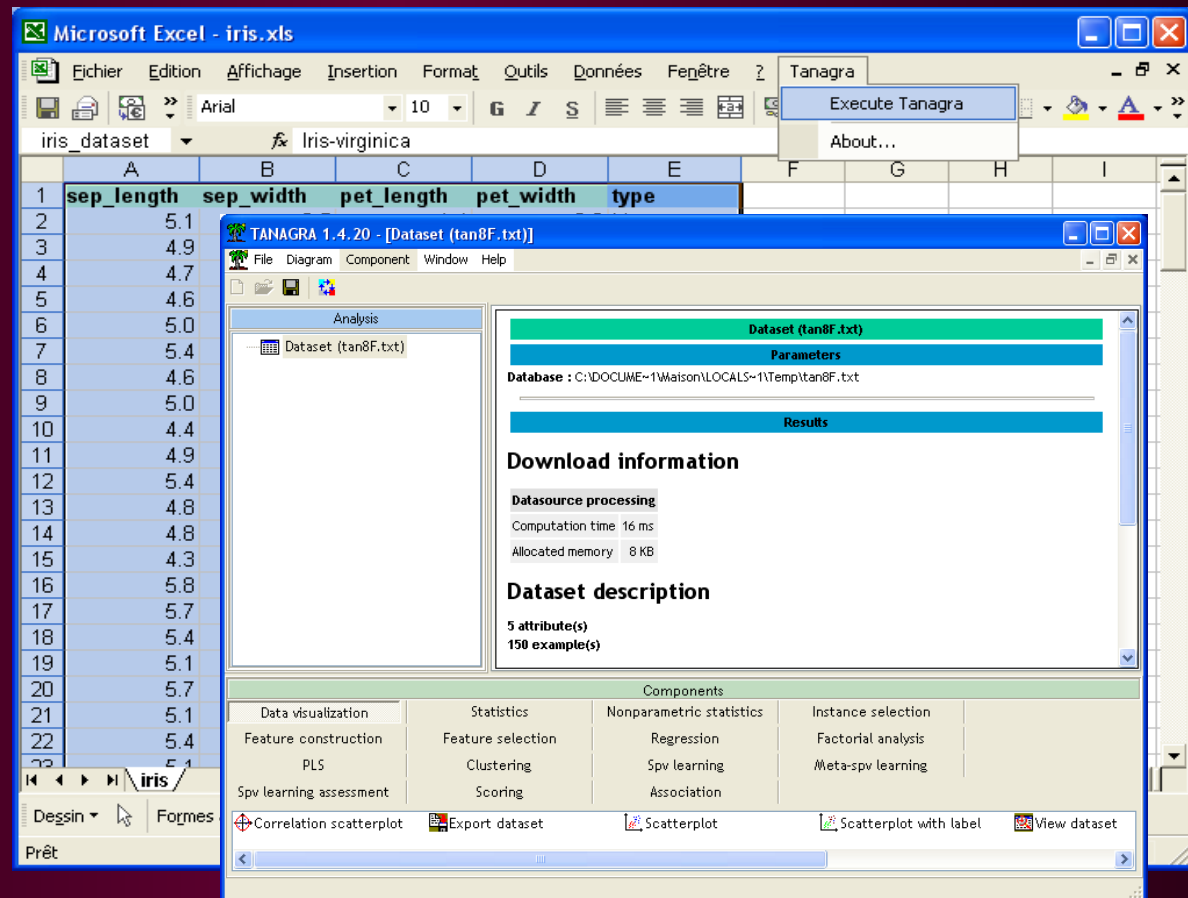
Importation de fichiers texte et format propriétaire

Texte avec séparateur tabulation est un standard reconnu (Tableur EXCEL)

Mais ça reste un problème récurrent (50% des questions sur SIPINA)

Refaire l'import à chaque fois

La transformation au format binaire est mal connu (mal documenté)



Interfaçage avec un tableur

Possibilité aussi de branchement externe sur un fichier texte ou un fichier XLS
On peut m  tre le fichier source sans refaire le traitement

Macro-compl  mentaire dans EXCEL (cf. enqu  te KDD)

Installation simple et automatisée

Tout doit fonctionner du premier coup



Tout doit être automatisé

L'utilisateur ne doit jamais avoir à intervenir à l'installation

Attention aux bibliothèques externes (SGBD, TCL/TK, PYTHON, etc.)

Choisir la configuration au pire cas

Réduire les bibliothèques externes

Bibliothèque externe compilée = dépendance accrue

Bibliothèque payante = pieds et poings liés (y compris sur les architectures)

Miser sur des versions stables et sources libres

Attention à la gestion des mises à jour

Mettre des exemples de traitements

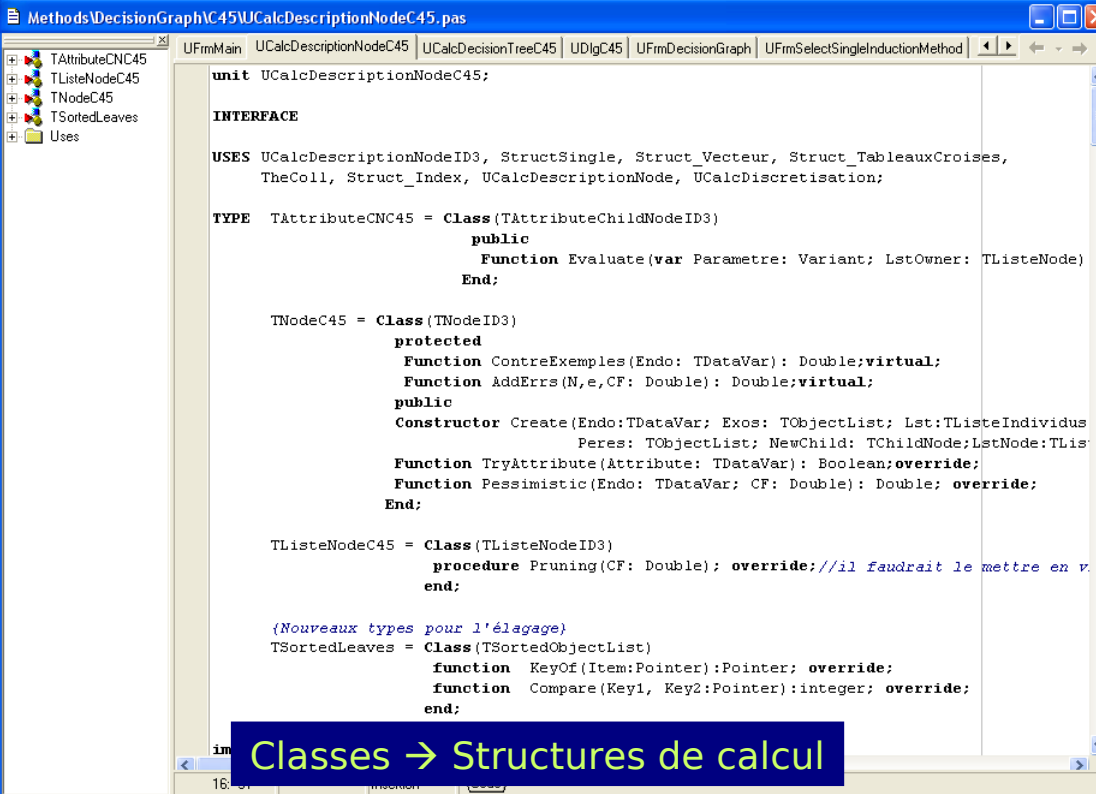
L'utilisateur lance toujours « pour voir » sans lire la documentation

4.b. TANAGRA - Développement

Implémentation

L'idée maîtresse est la standardisation poussée à l'extrême

Sipina



```
unit UC45DescriptionNodeC45;

INTERFACE

USES UC45DescriptionNodeID3, StructSingle, Struct_Vecteur, Struct_TableauxCroises,
    TheColl, Struct_Index, UC45DescriptionNode, UC45Discretisation;

TYPE TAttributeCNC45 = Class(TAttributeChildNodeID3)
    public
        Function Evaluate(var Parametre: Variant; LstOwner: TListeNode)
        End;

    TNodeC45 = Class(TNodeID3)
        protected
            Function ContreExemples(Endo: TDataVar): Double;virtual;
            Function AddErrs(N,e,CF: Double): Double;virtual;
        public
            Constructor Create(Endo:TDataVar; Exos: TObjectList; Lst:TListeIndividus
                Peres: TObjectList; NewChild: TChildNode;LstNode:TListeIndividus);
            Function TryAttribute(Attribute: TDataVar): Boolean;override;
            Function Pessimistic(Endo: TDataVar; CF: Double): Double; override;
        End;

    TListeNodeC45 = Class(TListeNodeID3)
        procedure Pruning(CF: Double); override;//il faudrait le mettre en v.
        end;

    {Nouveaux types pour l'élagage}
    TSortedLeaves = Class(TSortedObjectList)
        function KeyOf(Item:Pointer):Pointer; override;
        function Compare(Key1, Key2:Pointer):integer; override;
        end;

implementation
```

Classes → Structures de calcul

```
Methods\DecisionGraph\U45\UCalcDecisionTreeC45.pas
UFormMain | UCalcDescriptionNodeC45 | UCalcDecisionTreeC45 | UDlgC45 | UFormDecisionGraph | UFormSelectSingleInductionMethod
+ TArbreDecisionC45
+ Uses

unit UCalcDecisionTreeC45;
{Test pour l'instant}
{Le plus important est que l'on introduit l'élégage ici}

interface

USES UCalcArbreDecisionID3, StructSingle, TheColl, Classes, Forms,
    Struct_Index, Struct_Vecteur, Struct_TableauxCroises, SysUtils,
    ApprentissageGrid, IniFiles, UCalcComiteLearning;

TYPE TArbreDecisionC45 = Class(TArbreDecisionID3)
    Constructor Create(FN:ShortString; Endo: TDataVar; Exos: TObjectList
        ComitePere: TComiteLearning;
        Var Params: TMemIniFile; Section: ShortString; Showin
    procedure Learning(LstExamples: TListeIndividus; Editor: TApprent
    End;

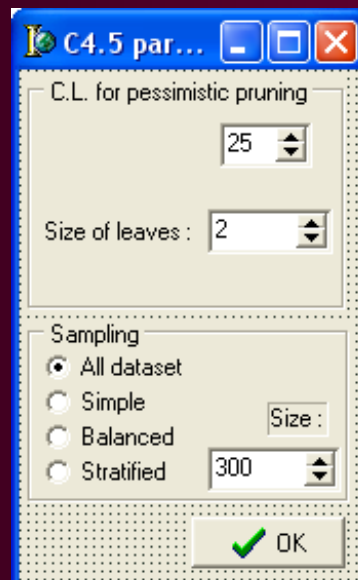
implementation

USES UCalcDescriptionNodeC45, UCalcDescriptionNode ,UBaseClasseApprentissage, Variants;

{Création de l'objet}
Constructor TArbreDecisionC45.Create(FN:ShortString; Endo: TDataVar; Exos: TObjectList;
    ComitePere: TComiteLearning;
    Var Params: TMemIniFile; Section: ShortString; Showing
Begin
    inherited Create(FN,Endo,Exos,ComitePere,Params,Section,FALSE,FicheMere);
    //détruire ce qui a été créé précédemment
    VarClear(Parametre);
    Nodes.Free;
end;
```

Classe → Instance de lancement des calculs

11: 28 | Insertion | \Code/



Classe → Fenêtre de paramétrage



Fenêtre de visualisation


```
UCALCParameters.pas
TAlgoParameter
TResultTest
Variables/Constante
Uses

Initialization
AlgoPrms:= Nil;
AlgoPrm:= TAlgoParameter.Create('');
{Liste de noms d'algos...}
BoitesDLGAlgos:= TStringList.Create;
//liste méthodes
With BoitesDLGAlgos Do
Begin
//arbres et graphes
Add('A limited search induction tree algorithm (Catlett - 1991)'+'+'+TFrmSimpleAlgo');
Add('ID3-IV (Quinlan - 1986)'+'+'+TFrmDlgID3');
Add('GID3 (Cheng, Fayyad, Irani & Qian - 1988)'+'+'+TFrmDlgGID3');
Add('ASSISTANT 86 (Cestnik, Kononenko & Bratko - 1986)'+'+'+TFrmDlgAssistant86');
Add('ChAID (Kass - 1980)'+'+'+TFrmDlgChAID');
Add('C4.5 (Quinlan - 1993)'+'+'+TFrmDlgC45');
Add
Add
Add
Add
Add
Add('Very fast C4.5 (Naive Bayes assumption)'+'+'+TFrmDlgFastC45');
//Analyse discriminante
Add('Linear discriminant analysis'+'+'+TFrmDlgLinearDA');
Add('Quadratic discriminant analysis'+'+'+TFrmDlgQuadraticDA');
//Réseaux de neurones
Add('Multi-Layer Perceptron'+'+'+TFrmDlgSimpleMLP');
Add('Single-Layer Perceptron'+'+'+TFrmDlgSingleLayerPerceptron');
Add('Multi-Layer Perceptron (Test Error rate control)'+'+'+TFrmDlgMLP_TERC');
//Induction de règles
Add('CN2 (Clark & Boswell, 1991)'+'+'+TFrmDlgPrmCN2');
Add('Improved CN2 (Rakotomalala, Feschet & Zighed, 1998)'+'+'+TFrmDlgPrmCN2Improved');
Add('C4.5 Rules (Quinlan, 1993)'+'+'+TFrmDlgC45Rules');
//TEBL
```

Enregistrer la méthode
La gestion des versions devenait cauchemardesque au fil du temps



```
D:\_Travaux\personal\Programmation\Tanagra\Source\Diagram\VML\Components\Spv\Learning\Spv\Algorithm\DecisionTree\UCalcSpvTreeC45.pas
UFrmMainForm | UCompSpvTreeC45 | UCalcSpvTreeC45

TYPE
//feuille
TSplitLeafSpvC45 = class(TSplitLeafSpv)
    end;

//split
TSplitAttributSpvC45 = class(TSplitAttributSpv)
    protected
    function    getClassSplitLeaf(): TClassSplitLeaf; override;
    function    ComputeGoodness(): double; override;
    function    ComputeAcceptSplit(): boolean; override;
    end;

//liste de splits
TLstSplitAttSpvC45 = class(TLstSplitAttSpv)
    protected
    function    getClassSplitAttribut(): TClassSplitAttribut; override;
    end;

//noeud de l'arbre
TMLTreeNodeSpvC45 = class(TMLTreeNodeSpv)
    private
    //erreur pessimiste (c'est le nombre d'erreur ici !!!) -- calculée lors de l'as
    FPessimisticErr: double;
    //calcul de l'écart à l'erreur pour avoir la borne -- taille sommet, contre-exe
    //extrait du livre de Quinlan, "Programs for Machine Learning.."
    function    addErrs(N,CE,CF: double): double;
    protected
    procedure    AssignConclusion(); override;
    function    isNoSplitNeeded(): boolean; override;
    function    getClassLstSplitAttributes(): TClassLstSplitAttributes; override;
    public
    //savoir si un noeud est prunable, i.e. a des enfants, et ce sont tous des feui
    function    isPrunable(): boolean;
    //erreur pessimiste
    property    pessimistic: double read FPessimisticErr;
    end;

//structure d'arbre
TMLTreeStructureSpvC45 = class(TMLTreeStructureSpv)
    protected
    function    getClassMLTreeNode(): TClassMLTreeNode; override;
    public
    //ce qui est spécifique à C4.5
    procedure    PostPruning(); override;
    end;

implementation
```

Classe → Structure de calcul

D:_Travaux\personal\Programmation\Tanagra\Source\Diagram\MLComponents\SvpLearning\SvpAlgorithm\DecisionTree\UCompSvpTreeC45.pas

UCompSvpTreeC45

```
TYPE
{générateur de composant C4.5}
TMLGCompSvpTreeC45 = class(TMLGCompSvpTree)
    public
        function    GetClassMLComponent: TClassMLComponent; override;
        end;

{le composant SvpTree C4.5}
TMLCompSvpTreeC45 = class(TMLCompSvpTree)
    protected
        function    getClassOperator: TClassOperator; override;
        end;

{l'opérateur C4.5}
TopSvpTreeC45 = class(TopSvpTree)
    protected
        function    getClassParameter: TClassOperatorParameter; override;
        function    getClassSvpLearning(): TClassCalcSvpLearning; override;
        end;

{paramètre de l'algo C4.5}
TopPrmSvpTreeC45 = class(TopPrmSvpTree)
    private
        //effectif d'admissibilité -- 5 par défaut
        FSizeMinLeaf: integer;
        //risque critique pour l'erreur pessimiste -- 0.25 par défaut
        FPLLevel: double;
    protected
        procedure   SetDefaultParameters(); override;
        function    CreateDlgParameters(): TForm; override;
    public
        function    getHTMLParameters(): string; override;
        procedure   LoadFromStream(prmStream: TStream); override;
        procedure   SaveToStream(prmStream: TStream); override;
        procedure   LoadFromINI(prmSection: string; prmINI: TMemIniFile); override;
        procedure   SaveToINI(prmSection: string; prmINI: TMemIniFile); override;
        property    SizeMinLeaf: integer read FSizeMinLeaf write FSizeMinLeaf;
        property    PLLevel: double read FPLLevel write FPLLevel;
        end;

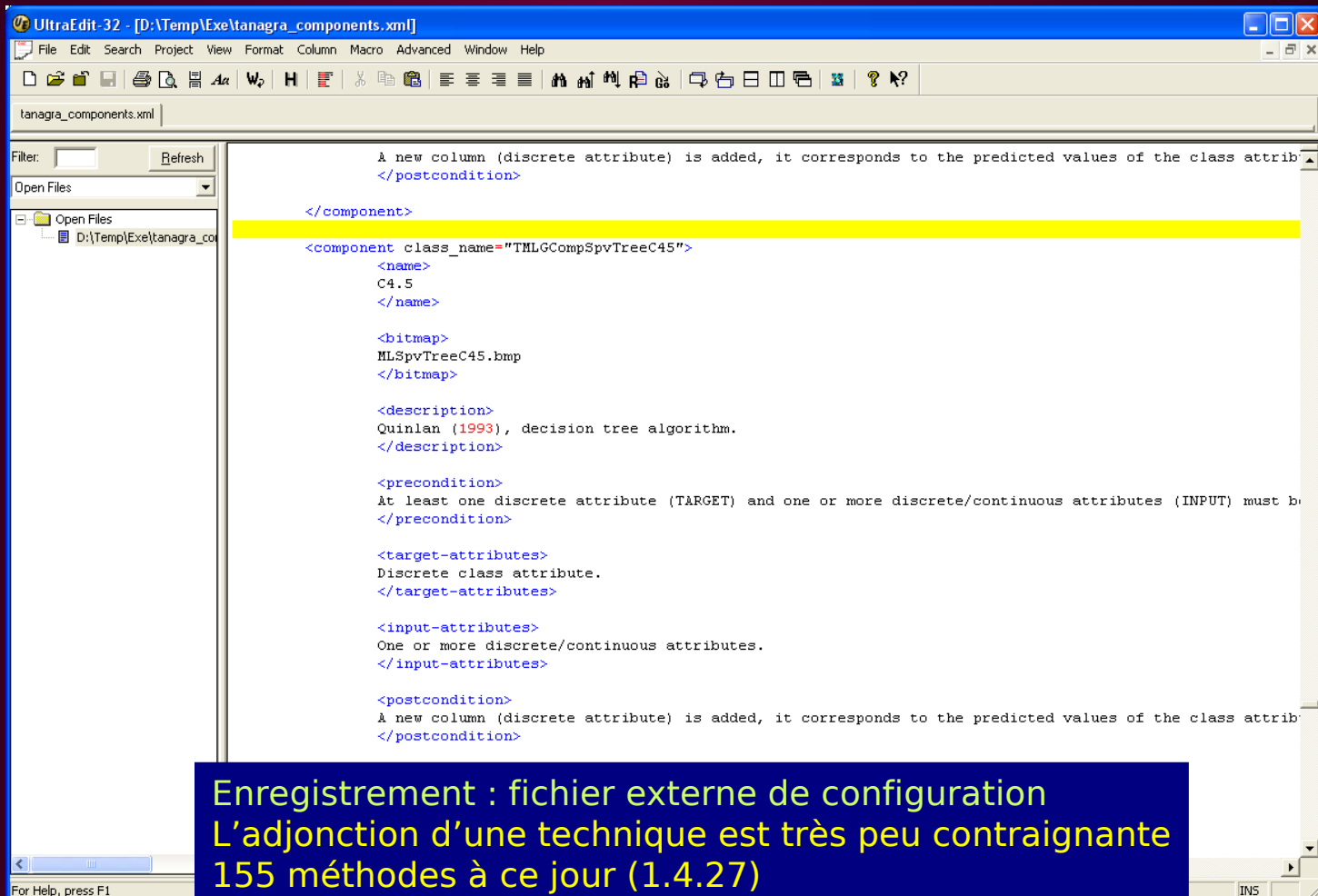
{algo d'arbre de décision supervisée}
TCalcSvpTreeC45 = class(TCalcSvpTree)
    protected
        function    getClassTreeStructureSvp(): TClassMLTreeStructure; override;
        end;

implementation
```

47: 35

Classe → Gestion de composant
Idée de « plug-in »





La solution idéale ?

L'application mère est une matrice qui gère et transmet les données

Les techniques sont des procédures programmées sous forme de bibliothèques externes

Mais des contraintes fortes

Organisation ultra-rigoureuse des protocoles

→ Passage des informations et des données

→ Affichage des résultats

Bref...

Souvent rédhibitoire, alors que l'objectif était d'offrir un outil modulaire

Intéressant si les plugins sont essentiellement des procédures de calculs qui renvoient des objets standardisés (ex. package R)

Et qu'une vraie communauté s'organise autour du logiciel

Implémentation

Quels outils de programmation ?

Spécifications

Outil libre (*ça coûte moins cher*)

Largement diffusé (*pour avoir des programmeurs*)

Avec une large bibliothèque de classes (*calculs, conteneurs, etc.*)

Qui permet de faire des interfaces agréables, simplement, rapidement

Pourquoi DELPHI pour Tanagra ?

A l'époque, DELPHI 6.0 PERSONAL était gratuite

Cours de DELPHI en L3 et M1 dans le département « Informatique – Statistique »

Accès aux anciennes bibliothèques de calculs, validées depuis longtemps déjà

Connaissance étendue des bibliothèques libres (Turbo Power, etc.)

Permet de faire des interfaces agréables, simplement, rapidement

Affinités personnelles...

...mais j'aurais du le faire en JAVA...?

4.c. TANAGRA - Promotion

Puisque le logiciel existe...
...autant le rendre disponible à d'autres.

Écrire un article de référence

Voilà toujours une publication de plus
Marquer le coup en annonçant le logiciel
C'est la référence que citeront les utilisateurs



Documenter le logiciel

Les méthodes qui sont intégrées
Leur mise en œuvre (sous forme de tutoriel)
Attention au danger du « manuel de référence – manuel de l'utilisateur » toujours en retard d'une version

Monter un site web attrayant (attractif)

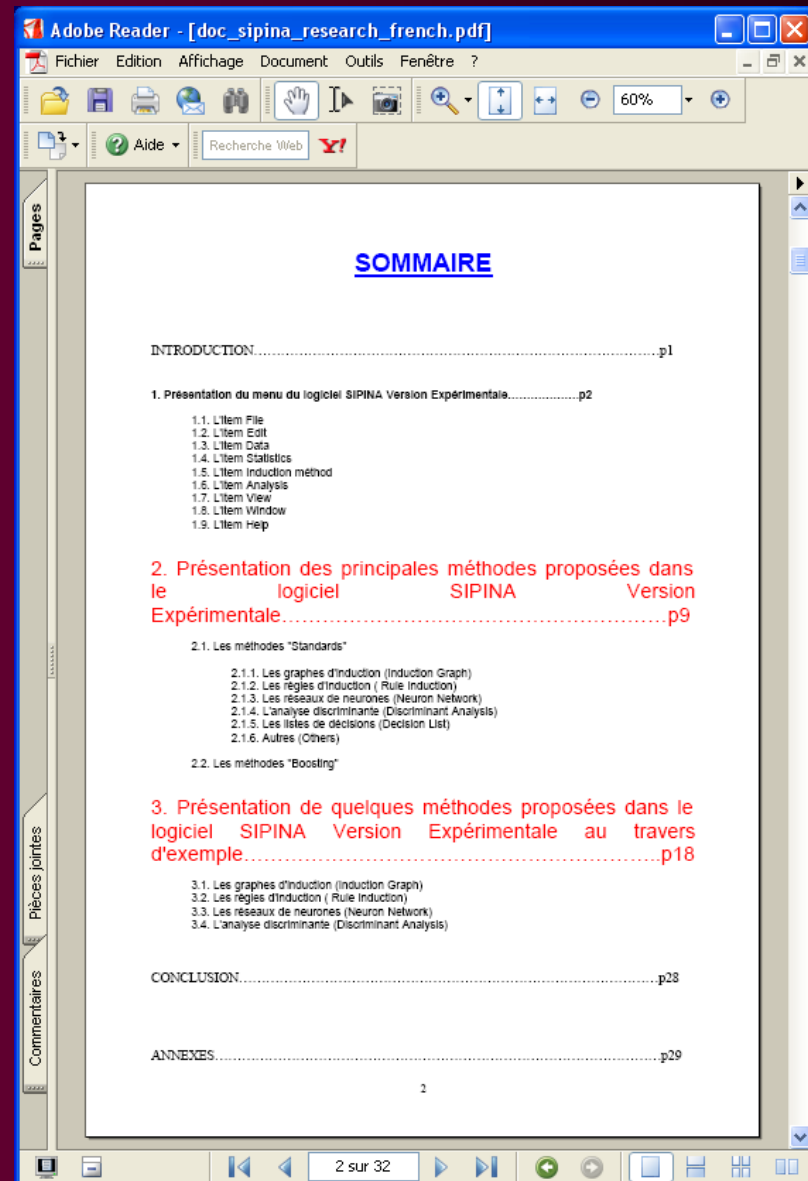
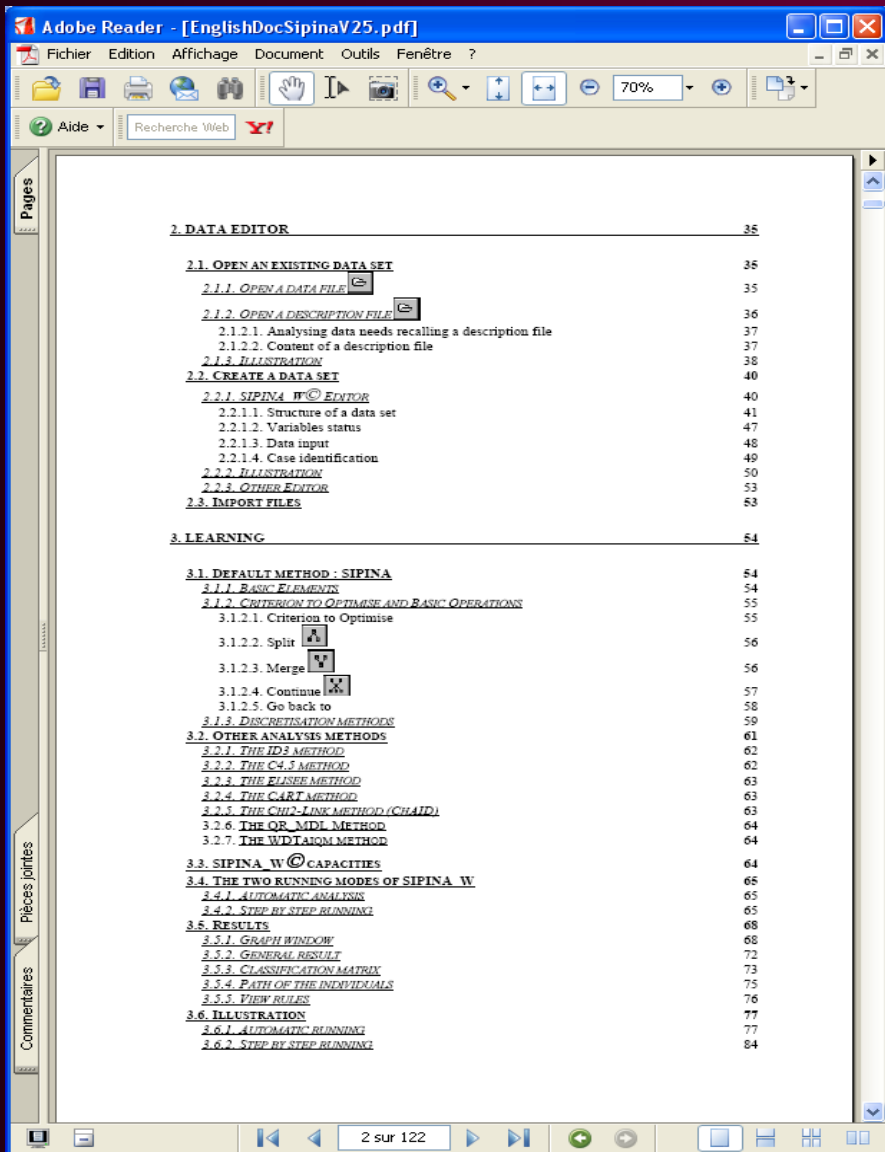
La visibilité internet est primordiale
Le téléchargement du logiciel n'est pas le seul enjeu

...et la promotion dans les conférences

Ateliers, démonstrations, contacts chercheurs, mailing-list, etc.

Documentation

Mettre à profit l'organisation du logiciel en composants



SIPINA - Une documentation classique difficile à faire vivre
Fastidieuses mises à jour
Copies d'écran à refaire avec les nouveaux menus

TANAGRA – Didacticiels « études de cas »

Fonctionnement/Organisation du logiciel jamais remis en cause
 → Ne jamais avoir à refaire les anciennes doc.

Se focaliser sur les méthodes et les aspects pédagogiques (lecture des résultats, les éléments de réflexion, etc.)

The screenshot shows the TANAGRA website interface. At the top, there is a navigation bar with icons for 'Présentation', 'Galerie', 'Caractéristiques', 'Didacticiels', and 'Téléchargement'. Below this is a table titled 'Didactiels - Statistique Exploratoire' with columns for 'Problématique', 'Opérateurs utilisés', 'Didacticiel', and 'Fichier'. The table lists several statistical methods with their corresponding software operators and document links.

Problématique	Opérateurs utilisés	Didacticiel	Fichier
Analyse en Composantes Principales (ACP) Lecture et mise en oeuvre de l'ACP. Les données et la description des résultats ont été alignés sur un exemple tiré de l'ouvrage de G. Saporta (« Probabilités, Analyse de Données et Statistique », Dunod, 2006, pages 177 à 181)	Dataset Define Status PCA		voitures
Analyse Factorielle des Correspondances (AFC) Lecture et mise en oeuvre de l'AFC. Les données et la description des résultats ont été alignés sur un exemple tiré de l'ouvrage de Lebart, Morineau et Piron, "Statistique Exploratoire Multidimensionnelle", Dunod, 2000 (pages 104 à 107).	Dataset Define Status CA		media
Analyse des Correspondances Multiples (ACM) Lecture et mise en oeuvre de l'ACM. Les données et la description des résultats ont été alignés sur un exemple tiré de l'ouvrage de M. Tenenhaus (« Méthodes Statistiques en Gestion », Dunod, 1996, pages 212 à 222)	Dataset Define Status MCA		races canines
Clustering -- HAC CAH MIXTE sur le fichier IRIS. Construction et interprétation des classes (clusters) avec le composant de caractérisation des groupes.	Dataset Define Status HAC Group characterization		iris
Clustering -- K-Means Construire des groupes avec un clustering, les confronter par la suite avec un regroupement naturel existant par ailleurs. Un exemple de "validation externe" en classification.	Multiple correspondance analysis K-Means Group characterization Cross-tabulation		vote
Classification de variables (VARCLUS) Mise en oeuvre de la classification de variables et lecture des résultats. Trois approches sont disponibles VARKMEANS, VARHCA, VARCLUS. La présentation est inspirée de la lecture de l'ouvrage de J.P. Nakache et J. Confais, "Approche Pragmatique de la Classification", Editions TECHNIP, 2005, chapitre 9. Une autre référence est le manuel de présentation d'un logiciel bien connu => http://www2.stat.unibo.it/Manuali\$as/stat/chap68.pdf	Dataset Define Status VARKMEANS VARHCA VARCLUS		crime dataset

Documentation - Améliorer l'organisation des didacticiels

Tutoriels Tanagra pour le Data Mining - Windows Internet Explorer fourni par Yahoo! France

http://tutoriels-data-mining.blogspot.com/

Fichier Edition Affichage Favoris Outils ?

Google

Tutoriels Tanagra pour le Data Mining

RECHERCHER LE BLOG | MARQUER LE BLOG | Blog suivant»

Créer un blog | Connexion

Tutoriels Tanagra pour le Data Mining

Ce blog recense mes didacticiels pour Tanagra rédigés à ce jour, accessibles dans la section Didacticiels du site web du logiciel (**le transfert est en cours...**). Les tutoriels sont organisés en catégories. On dispose des fonctionnalités de recherche par mots-clés d'un blog. Chaque article est accompagné d'un texte de présentation, d'une liste de mots-clés, du lien vers les données, du lien vers le didacticiel lui-même (document pdf), d'une ou plusieurs références bibliographiques lorsque cela est nécessaire.

Ricco Rakotomalala

JEUDI 3 AVRIL 2008

Tanagra : un logiciel open source

Ce document correspond à la première présentation du logiciel Tanagra en janvier 2004, à l'occasion d'un séminaire au sein du laboratoire ERIC.

L'idée est de décrire les enjeux et les contraintes qui pèsent sur les spécifications et la réalisation de Tanagra. Nous essayons de cerner les utilisateurs que l'on vise, les traitements types dans lesquels le logiciel peut être mis en avant, etc.

Un aspect très important est abordé lors de cet exposé : la nécessité pour les chercheurs de rendre accessibles les codes sources des prototypes servant aux publications. En effet, les développements théoriques sont souvent accompagnés d'expérimentations. Il est très important que les chercheurs puissent les reproduire, les modifier pour évaluer l'impact de tel ou tel paramétrage, les comparer avec d'autres implémentations d'un même algorithme, voire de les vérifier. Cela n'est possible que si le logiciel est en « open source ». D'une certaine manière, la publication du code sert aussi à valider les articles scientifiques.

Lien : slides_tanagra.pdf

Publié par Tanagra le 3.4.08
Libellés : Tanagra - Fonctionnalités

MERCREDI 2 AVRIL 2008

Supports et tutoriels

- Page principale du blog
- Cours Data Mining
- Portail Data Mining

Logiciels

- Site du logiciel Tanagra
- Téléchargement Tanagra
- Site du logiciel Sipina

Catégories des tutoriels

- Analyse discriminante (6)
- Analyse factorielle (11)
- App. Supervisé - Scoring (25)
- Arbres de décision (13)
- Classification - Clustering (7)
- Construction de variables (5)
- Importation des données (8)
- Règles d'association (5)
- Régression (5)
- Régression logistique (4)
- Régression PLS (2)
- Sipina (10)
- Statistiques et tests (4)
- Sélection de variables (5)
- Tanagra - Fonctionnalités (6)
- Tanagra et les autres (8)

Plusieurs possibilités d'indexation
(et donc de recherche)

- Catégories
- Mots-clés
- Références en ligne (*cours, etc.*)
- Références bibliographiques (*ouvrages, articles de référence, etc.*)

Documentation

Aller plus loin encore en proposant des supports de cours

Supports de cours -- Data Mining - Windows Internet Explorer fourni par Yahoo! France

http://eric.univ-lyon2.fr/~ricco/cours/supports_data_mining.html

support de cours data mining

Supports de cours -- Data Mining

Portails de e-learning, logiciels, autres supports en ligne, etc.

Statistique descriptive Description statistique, distribution empirique, indicateurs et graphiques.							
Probabilités et Statistique Analyse combinatoire, Théorie des Probabilités, Lois de Probabilités d'usage courant, Test d'adéquation à une loi.							
Statistique Inférentielle Estimation ponctuelle. Estimation par intervalle. Théorie des tests. Quelques tests paramétriques usuels.							
Biostatistique Probabilités et Statistiques. Statistique inférentielle : Estimation ponctuelle et par intervalle, Théorie des tests, etc... à l'usage des biologistes.					-	-	
Test de normalité Test statistique d'adéquation à la loi normale (normality test) : test de Shapiro Wilk, test de Lilliefors, test d'Anderson-Darling, test de D'Agostino, test de Jarque-Bera. Test de symétrie des distributions : test basé sur le coefficient d'asymétrie, test de Wilcoxon, test de Van der Waerden.							
Corrélation et corrélation partielle Corrélations linéaires, corrélations croisées, tests de significativité. Corrélations partielles.							
Mesures d'association pour variables nominales Test d'indépendance du KHI-2. Mesures dérivées du KHI-2 (T de Tschuprow, c de Cramer...). Mesures asymétriques d'association (PRE mesures) : Lambda et Tau de Goodman & Kruskal, U de Theil. Eléments spécifiques aux tableaux 2 x 2 : Q de Yule, Odds-ratio, Risque relatif, correction de Yates. Coefficient de concordance pour variables nominales : Kappa de Cohen, Kappa de Fleiss, Kappa généralisé. Mesures d'association pour les variables ordinales (Gamma de Goodman et Kruskal, Tau-b et Tau-c de Kendall, Gamma de Sommers).							
Tests non paramétriques - Comparaison de populations et étude des dépendances Test de Mann-Whitney, test de Kruskal-Wallis, rho de Spearman, tau de Kendall, coefficient de concordance de Kendall, test des signes, test de Wilcoxon pour échantillons appariés, tests des séquences de Wald-Wolfowitz et Mood							

Ricco Rakotomalala - Université Lyon 2

Description approfondie des techniques

Liens externes, autres descriptions

Fichiers de données

Didacticiel
TANAGRA



Écriture du cahier de charges

Janvier 2003

Plusieurs prototypes développés (C++, Java, Delphi)

Début du développement

Juillet 2003 → Moteur interne figé depuis fin août 2003

Création du site web et mise en ligne

Janvier 2004 (~25 visiteurs par jour sur 2004)

Techniques implémentées (Sept. 2008 – Version 1.4.27)

155 méthodes stat., exploratoires et data mining

Documentation

~100 didacticiels en français, 70 en anglais

Visites sur le site web Tanagra exclusivement – Janv. À Sept. 2008

~155 par jour en moyenne

(à titre de comparaison, bilan sur la période sept. à déc. 2005 : 70 visiteurs par jour)

Virage important début 2007 : mise en ligne de supports de cours

Visites sur le domaine (fév. à août 2008)



Quelles pages ?

	Titre de la page	Pages vues ↓
1.	Portail DATA MINING	16 701
2.	Tanagra FR	13 895
3.	Supports de cours -- Data Mining	13 643
4.	Tanagra EN	13 431
5.	Cours econometrie	12 464

Quels visiteurs ? (Monde académique en majorité)

	Niveau de détail : Pays/Territoire ↓	Visites ↓
1.	France	28 819
2.	Algeria	4 836
3.	Morocco	4 117
4.	United States	2 808
5.	Tunisia	2 705
6.	Canada	1 682
7.	Belgium	1 525
8.	Brazil	1 428
9.	India	1 141
10.	United Kingdom	948
11.	Switzerland	909
12.	Germany	896
13.	Italy	814
14.	Ivory Coast	805
15.	Senegal	621
16.	Spain	518
17.	Cameroon	490
18.	Burkina Faso	472
19.	Portugal	406
20.	Turkey	382
21.	Vietnam	326
22.	Indonesia	303
23.	Madagascar	290
24.	China	286
25.	Poland	266

5. TANAGRA - Traitements

Arbre de décision

Traitement du fichier Wave de Breiman

Scoring

Détection des clients « fiables » en Banque

Mesures d'association pour les variables nominales

Consommation de véhicules

6. Et les autres...

WEKA

La référence

Référence mondiale
Bibliothèque de méthodes très riche
Essentiellement machine learning
Ajout de méthodes
Ajout de classes + recompilation
Accès compliqué à l'interface
Atout : appel des classes en externe

Mode explorer

Weka 3.5.6 - Explorer

Program Applications Tools Visualization Windows Help

Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose **J48** -C 0.25 -M 2

Test options

Use training set

Supplied test set Set...

Cross-validation Folds

Percentage split %

More options...

(Nom) class

Start Stop

Result list (right-click for options)

15:18:59 - trees.J48

Classifier output

```
=== Classifier model (full training set) ===  
  
J48 pruned tree  
-----  
petalwidth <= 0.6: Iris-setosa  
petalwidth > 0.6  
| petalwidth <= 1.7  
| | petalwidth <= 4.9: Iris-versicolor  
| | | petalwidth > 4.9  
| | | | petalwidth <= 1.5: Iris-versicolor  
| | | | petalwidth > 1.5: Iris-virginica  
| | | | | petalwidth > 1.7: Iris-virginica  
  
Number of Leaves : 5  
Size of the tree : 9  
  
Time taken to build model: 0.05  
  
=== Stratified cross-validation ===  
=== Summary ===  
  
Correctly Classified Instances  
Incorrectly Classified Instances  
Kappa statistic  
Mean absolute error
```

Status
OK

Mode Knowledge flow

Weka 3.5.6 - KnowledgeFlow

Program Applications Tools Visualization Windows Help

KnowledgeFlow

DataSources DataSinks Filters Classifiers Clusterers Associations Evaluation Visualization

Visualization

Data Visualiser Scatter PlotMatrix Attribute Summariser Model PerformanceChart Text Viewer Graph Viewer Scrip Chart

Knowledge Flow Layout

```
graph LR  
iris[iris] -- data3e --> TrainTestSplitMaker[TrainTest SplitMaker]  
TrainTestSplitMaker -- trainin testSe --> J48[J48]  
J48 -- graph --> GraphViewer[GraphViewer]  
J48 -- batchClassifier --> ClassifierPerformanceEvaluator[Classifier Performance Evaluator]  
ClassifierPerformanceEvaluator -- text --> TextViewer[Text Viewer]
```

Status
Done.

Text Viewer

Result list

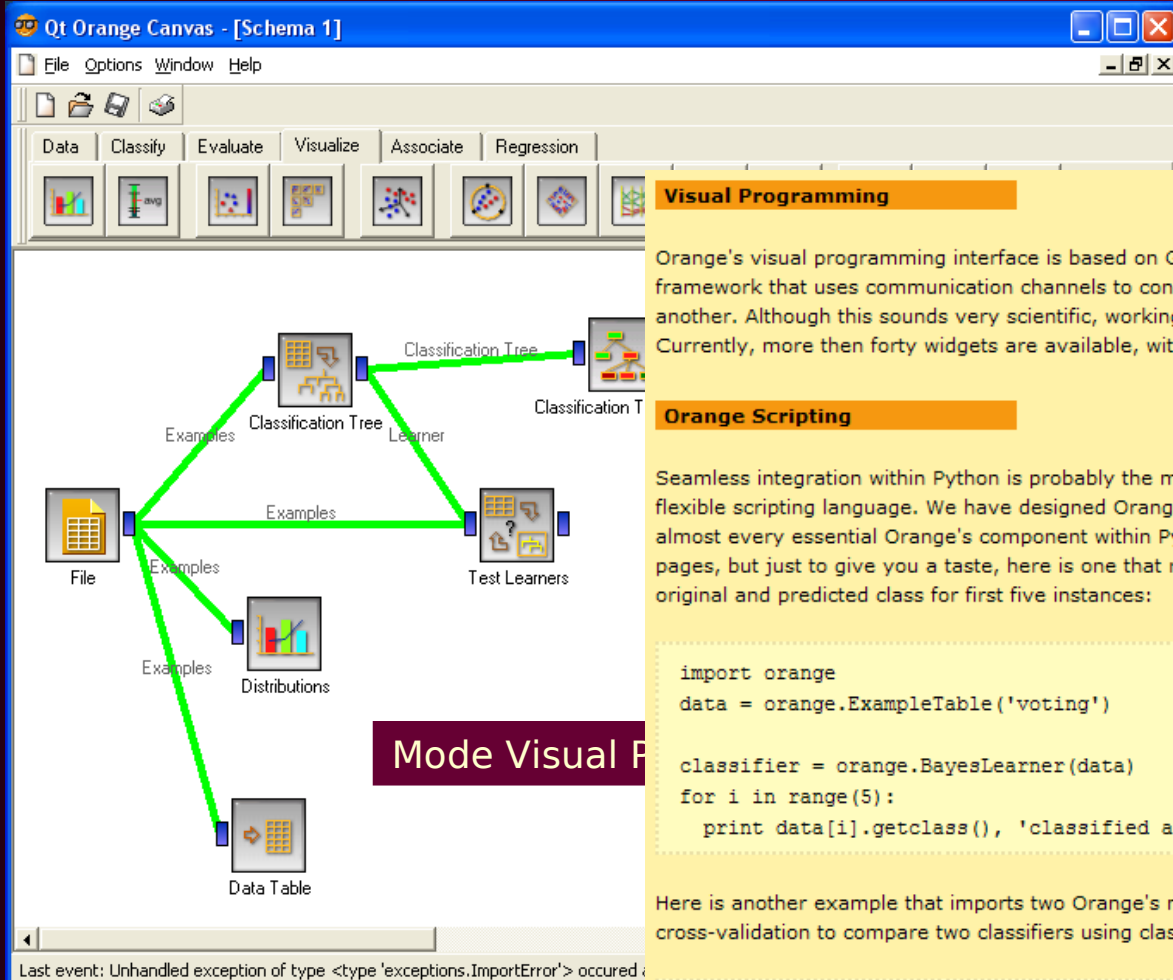
15:25:58 - J48

```
=== Confusion Matrix ===  
  
a b c <-- classified as  
15 0 0 | a = Iris-setosa  
0 19 0 | b = Iris-versicolor  
0 2 15 | c = Iris-virginica
```

ORANGE

Interface agréable, système de plugins, langage interprété

Interface agréable et performante
Outils graphiques
Essentiellement machine learning
Atout (1) : méthode = plugins
Atout (2) : interpréteur python



Mode Visual P

Visual Programming

Orange's visual programming interface is based on GUI components we call Orange Widgets, and a signalling framework that uses communication channels to connect widgets and tokens to pass the data from one widget to another. Although this sounds very scientific, working with widgets in the Orange Canvas is simple as point-and-click. Currently, more than forty widgets are available, with more coming out every week.

Orange Scripting

Seamless integration within Python is probably the most important feature of Orange. Python is a great and very flexible scripting language. We have designed Orange to be fully accessible within Python and are trying to expose almost every essential Orange's component within Python. We provide a number of [example scripts](#) on Orange's web pages, but just to give you a taste, here is one that reads the data file, builds a naive Bayesian classifier and outputs original and predicted class for first five instances:

Mode Script

```
import orange
data = orange.ExampleTable('voting')

classifier = orange.BayesLearner(data)
for i in range(5):
    print data[i].getclass(), 'classified as', classifier(data[i])
```

Here is another example that imports two Orange's modules (orngTest and orngStat), reads the data, and uses cross-validation to compare two classifiers using classification accuracy and Brier score:

```
import orange, orngTest, orngStat
data = orange.ExampleTable('voting')

bayes = orange.BayesLearner()
tree = orange.TreeLearner()

results = orngTest.crossValidation([bayes,tree], data, folds=5)
print 'Classification accuracy: ', orngStat.CA(results)
print 'Brier Score: ', orngStat.BrierScore(results)
```

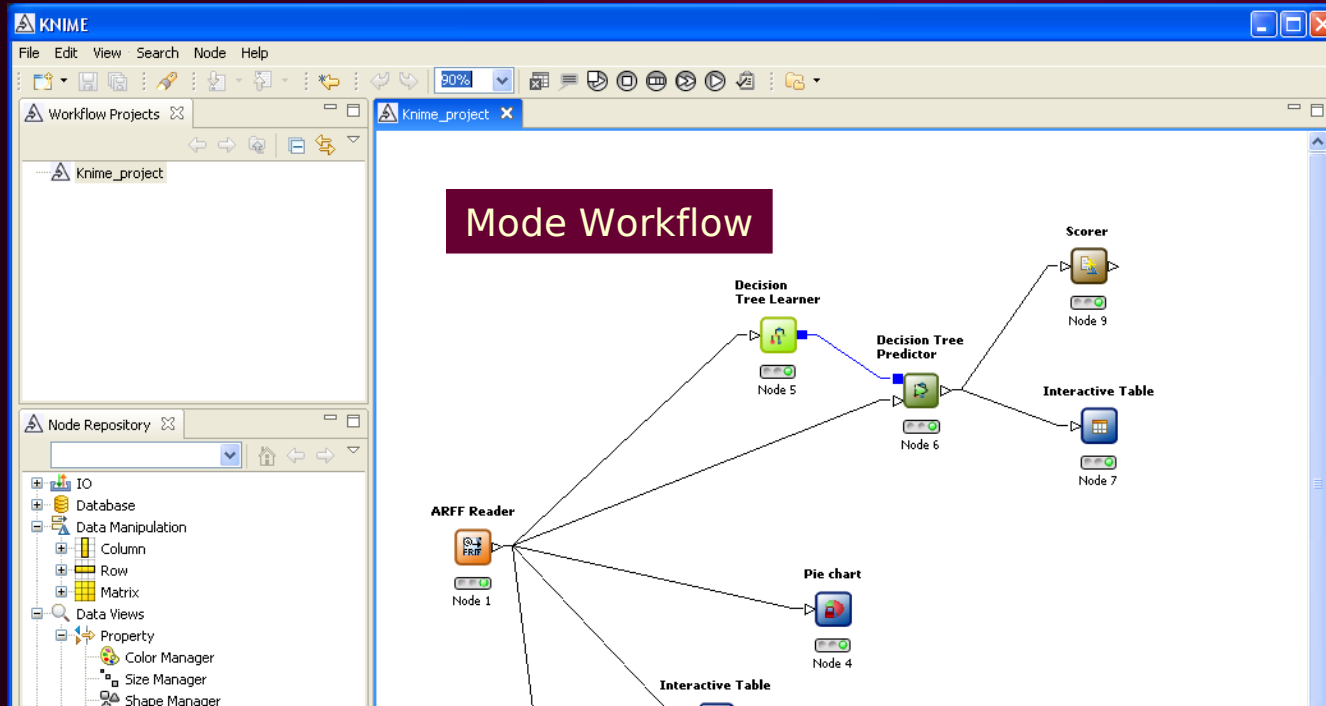
KNIME

Interface agréable, plugins WEKA et R...

Interface agréable et performante
Outils graphiques
Outils de manipulation de données
Pas de programmation possible

Atout (1) : méthode = plugins (API)

Atout (2) : accès aux méthodes WEKA et R (!)



KNIME base version already incorporates over 100 processing nodes for data I/O, preprocessing and cleansing, modelling, analysis and data mining as well as various interactive views, such as scatter plots, parallel coordinates and others. It includes all analysis modules of the well known Weka data mining environment (<http://www.cs.waikato.ac.nz/ml/weka/>) and additional plugins allow R-scripts (www.r-project.org) to be run, offering access to a vast library of statistical routines.

KNIME is based on the Eclipse platform and, through its modular API, easily extensible. Custom nodes and types can be integrated within hours enabling KNIME to be used not only in production environments but also for teaching and research prototyping. If you would like to read a more detailed description of the software, please download our White Paper as a PDF file [here](#).

7. *A posteriori...*

Choix d'architecture fondamentale

Objectif TANAGRA, empiler des méthodes

- Choisir une architecture qui permet de faciliter l'ajout de méthodes (composants)
- Minimiser la programmation « annexe » (interfaces visuelles, accès externes, etc.)

Communauté de développeurs

Il faut une organisation très rigoureuse (et jouer le rôle de « chef de projet » ?)

- c'est un travail à plein temps

Documentation, primordiale pour la diffusion

Sur les méthodes, sur la mise en œuvre

- Centrer la documentation sur les méthodes et non sur le logiciel

Élargir le public : chercheur, étudiants, personnes d'autres domaines, etc.