

Module Intégratif

Extraction de connaissances dans les
bases de données

Environnement Logiciel

Environnement de travail

- Orange (<http://orange.biolab.si/>)
 - Python 2.7 (<http://www.python.org/download/releases/2.7/>)
 - Bibliothèques additionnelles
 - Matplotlib (<http://matplotlib.sourceforge.net/>)
 - Scipy (<http://www.scipy.org/>)
 - Tous ces éléments sont libres
 - Vous pouvez donc les télécharger et les installer sur vos postes personnels !
-

Orange

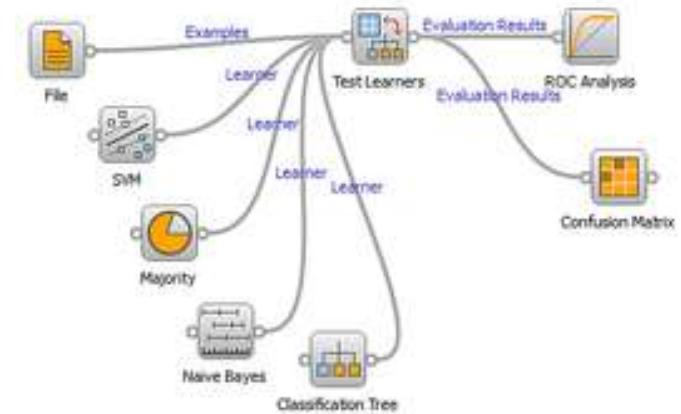
- Outil intégré et libre
 - Téléchargeable à l'adresse : <http://orange.biolab.si/>
 - Librairie pour
 - la manipulation de données,
 - Le Data mining,
 - l'apprentissage supervisé,
 - Etc.
 - Codée en C++, Surcouche en Python
 - Programmation graphique et scripts
-

Programmation graphique

- Construire un logiciel en programmation graphique
 - dessiner l'ordinogramme du logiciel
 - assembler des icônes



Un schéma



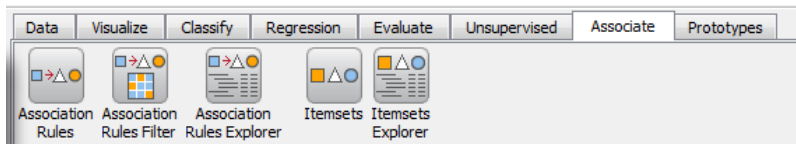
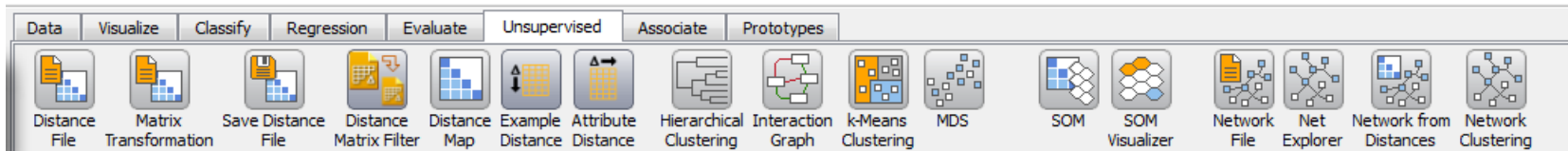
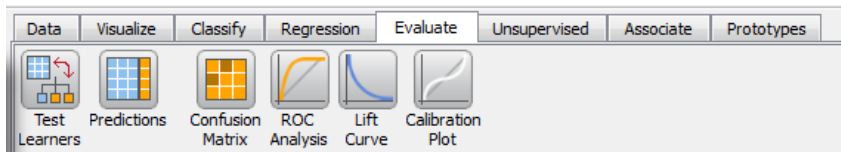
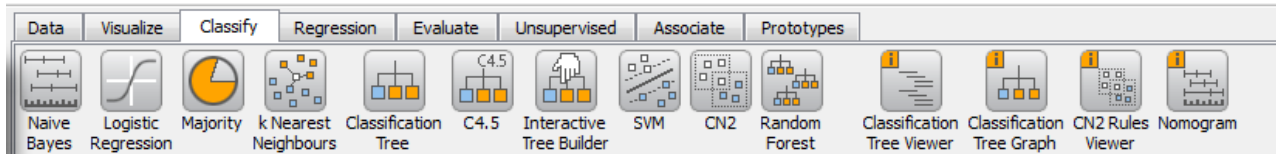
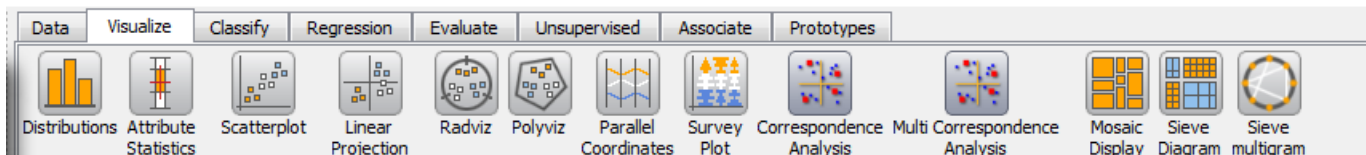
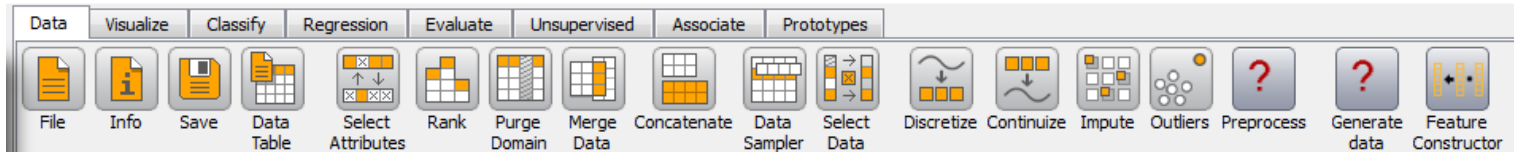
- Les icônes (widgets) représentent des éléments de programmes permettant de faire des traitements particuliers
 - ouvrir un fichier, construire un graphique, ...

Programmation graphique

- Programmation plus intuitive
 - Comme une fonction les widgets ont des entrées et des sorties
 - Une boîte s'exécute lorsque ses données d'entrées sont disponibles donc lorsque la boîte qui la précède a terminé son traitement et rendu son résultat
 - Programmation par flux de données
 - Animations d'exécution
 - Structure parallèle
-

Orange : Widgets

Un catalogue fournit et ordonné en onglets



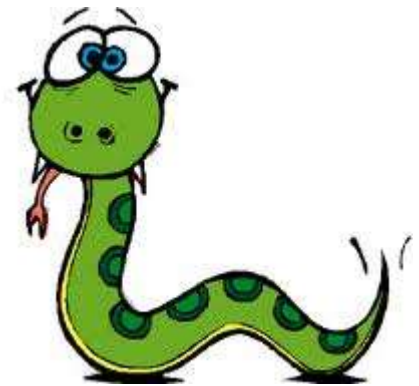
Orange : Widgets

- Aide en ligne :
 - <http://orange.biolab.si/doc/catalog/>
- Possibilité de script en Python pour utilisation et compléter les fonctionnalités proposées



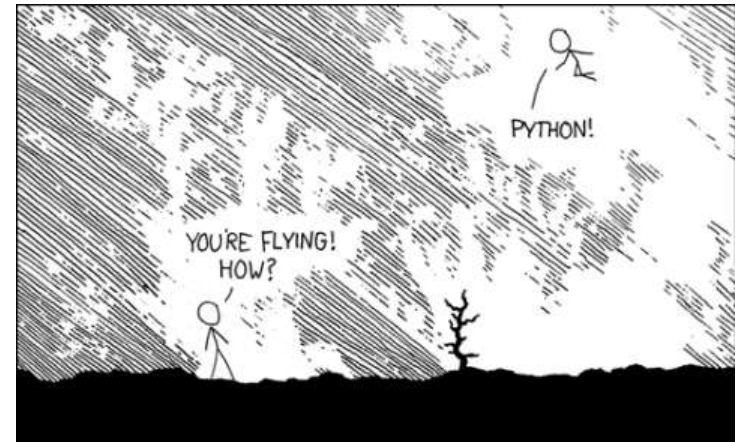
Python

Premiers pas ...

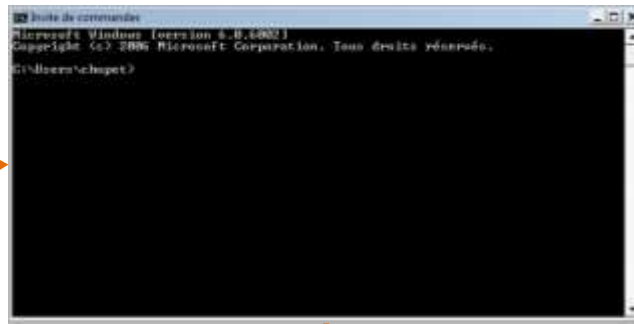
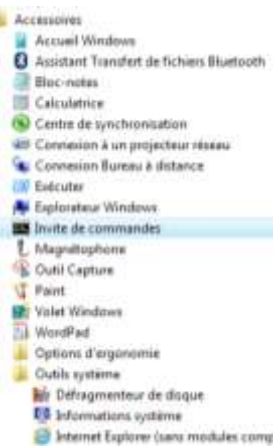


Pourquoi Python ?

- Python est un langage
 - performant, très intuitif et facile d'accès (pseudo-code)
 - Sa syntaxe permet d'obtenir un code simple et aéré.
 - multiplateforme et gratuit
- Python possède un shell (interpréteur)
 - tester directement les instructions en ligne de commande.
- Python est développé et maintenu par une communauté très dynamique
 - Nombreuses bibliothèques adaptées à des besoins spécifiques
 - Calcul numérique
 - Analyse de données
 - Traitement d'images
 - Bioinformatique,
 - ...
- De plus en plus utilisé
 - Industrie, secteur public, recherche, ...

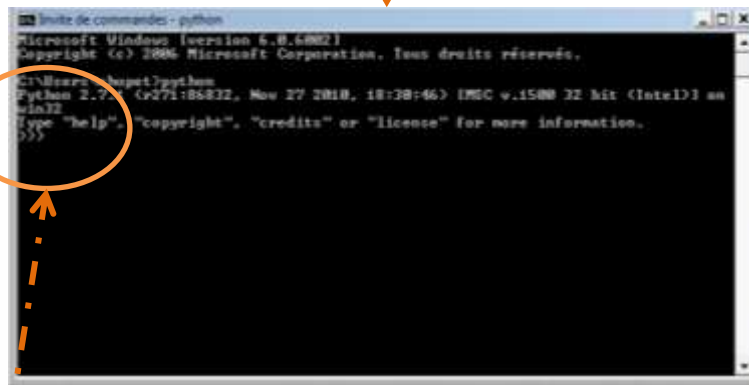


Python et l'Invite de commandes



Fenêtre de commandes

Tapez python et valider



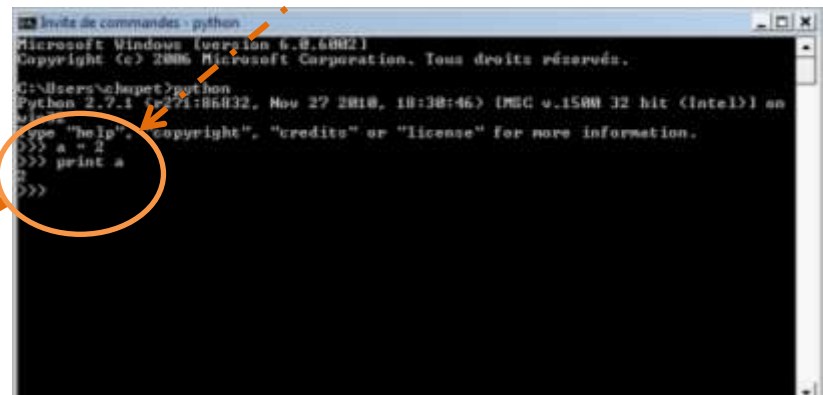
Les résultats des instructions est fourni directement par validation des instructions

Dans le menu des programmes, sélectionnez : « Invite de commandes »

L'interpréteur Python est lancé.

Il attend vos instructions

>>>



Outils en ligne

- <http://shell.appspot.com/>

Interactive server-side Python shell for [Google App Engine](#). ([source](#))

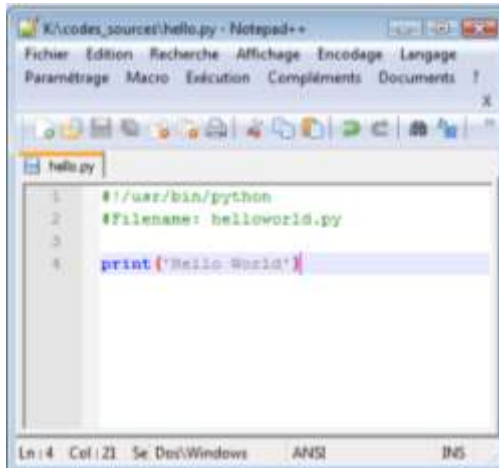
```
Google App Engine/1.5.0
Python 2.5.2 (r252:60911, Mar 17 2011, 15:16:30)
[GCC 4.3.1]

>>> print 'Hello World'
Hello World
```

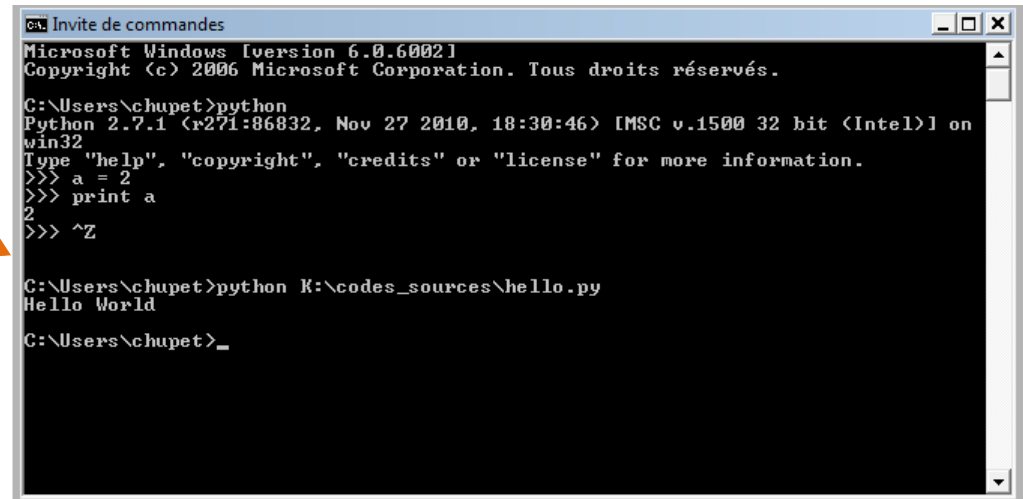
```
>>>
```

```
..
```

Exécuter des scripts Python



```
K:\codes_sources\hello.py - Notepad++
Fichier Edition Recherche Affichage Encodage Langage
Paramétrage Macro Exécution Compléments Documents ?
hello.py
1 #!/usr/bin/python
2 #filename: helloworld.py
3
4 print('Hello World')
```



```
ca. Invite de commandes
Microsoft Windows [version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation. Tous droits réservés.

C:\Users\chupet>python
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> a = 2
>>> print a
2
>>> ^Z

C:\Users\chupet>python K:\codes_sources\hello.py
Hello World

C:\Users\chupet>_
```

Quand vous enregistrez un script python, celui-ci doit porter l'extension **.py**

Structure, Syntaxe d'un programme Python

```
# -*- encoding: iso-8859-15 -*-  
  
from Tkinter import *  
  
def factorielle(event):  
    n = int(input1.get())  
    i = 1  
    while n > 0:  
        i *= n  
        n = n - 1  
    resultat.configure(text="Le résultat est : " + str(input1.get()) + "! = " + str(i))  
  
fenetre = Tk()  
  
textel = Label(fenetre, text='Entrez le nombre :', fg='black')  
textel.grid(row = 0, column = 0, sticky = W)  
  
input1 = Entry(fenetre, bg='white', fg='red')  
input1.bind("<Return>", factorielle)  
input1.grid(row = 1, column = 1, sticky = W+E, ipadx = 5, ipady = 5)  
  
resultat = Label(fenetre)  
resultat.grid(row = 2, column = 1, sticky = W+E, ipadx = 5, ipady = 5)  
  
bouton2 = Button(fenetre, text='Quitter', command=fenetre.quit)  
bouton2.grid(row = 3, column = 2, sticky = E)  
  
fenetre.mainloop()
```

Déclaration de l'encodage

Importation des modules nécessaires

Définition d'une
fonction qui sera
utilisée plus tard

Appel de la fonction factorielle
définie avant

Corps principal du programme
(ici construction d'une interface graphique pour la fonction factorielle)

Syntaxe très légère

L'indentation détermine les
début et fin de blocs

Un entête de bloc se
termine par « : »