

## Indexation :

### 1. Extraction automatique de termes

```
>>> ExpReg = nltk.RegexpTokenizer('(?:[A-Za-z]\.|\d+(?:\.\d+)?|\w+(?:-\w+)*')
```

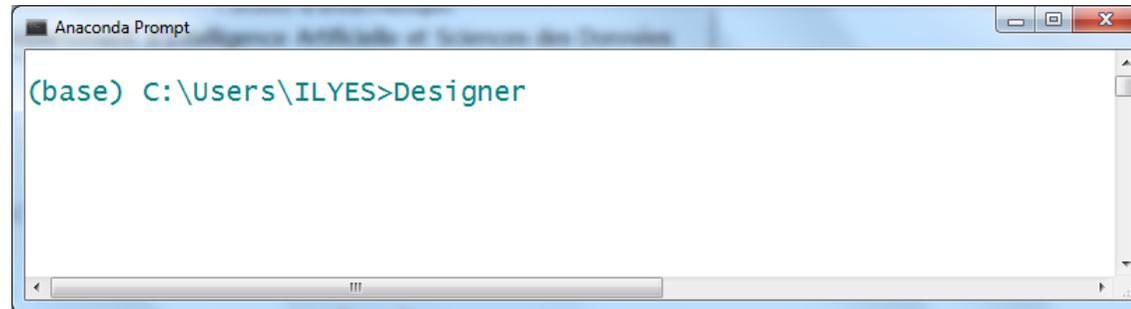
### 5. Pondération des termes normalisés

$$poids(t_i, d_j) = \left( \frac{freq(t_i, d_j)}{Max(freq(t, d_j))} \right) * \log \left( \frac{N}{n_i} + 1 \right)$$

IHM :

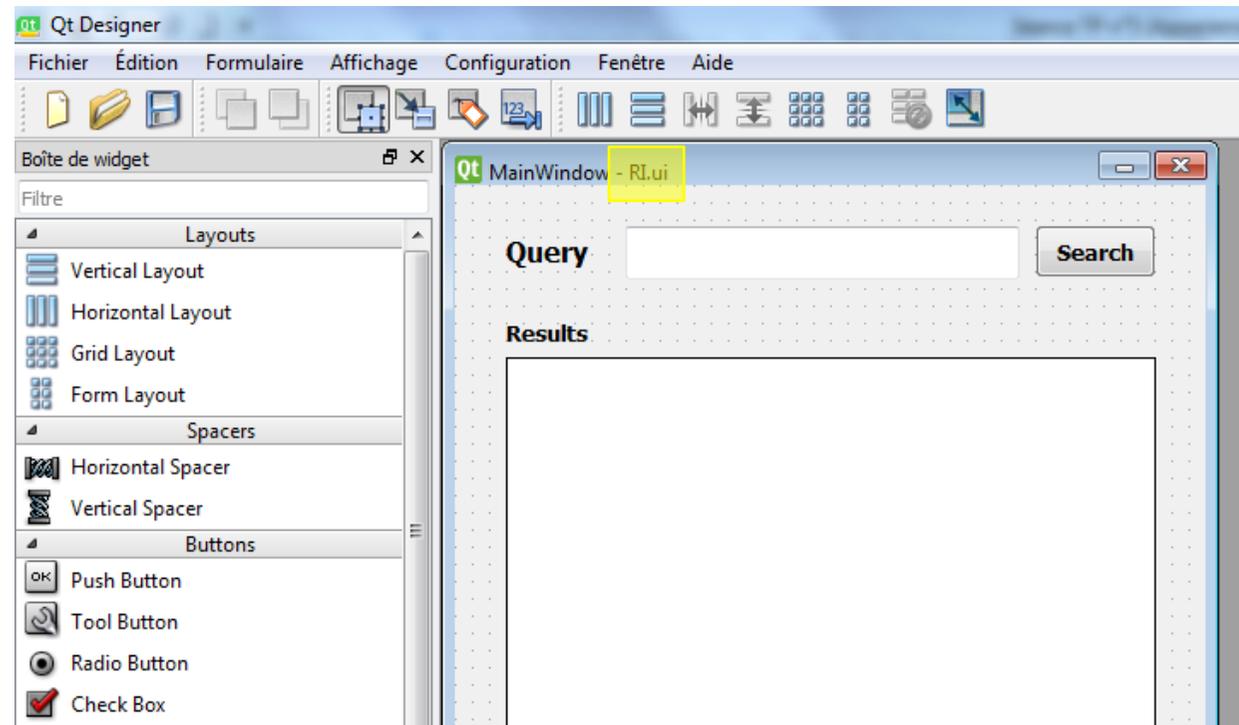
## 1. Qt Designer

>>> Designer



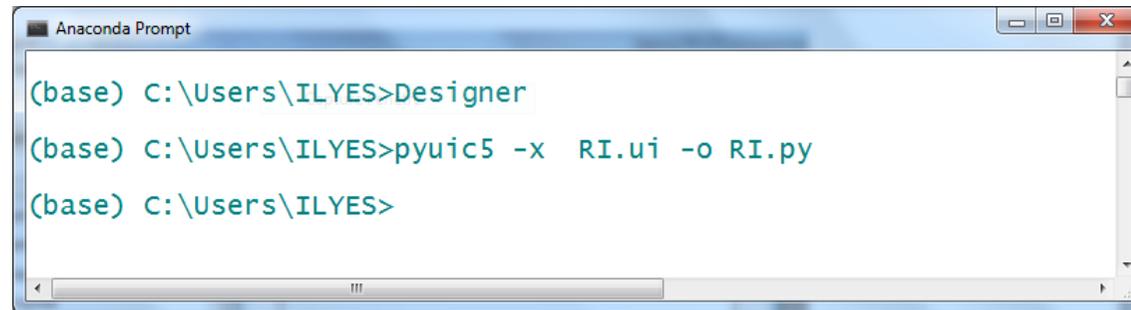
**Extension du fichier  
sur Qt Designer :**

<nom>.ui



## 2. Convertir le fichier <nom>.ui en <nom>.py

```
>>> pyuic5 -x RI.ui -o RI.py
```

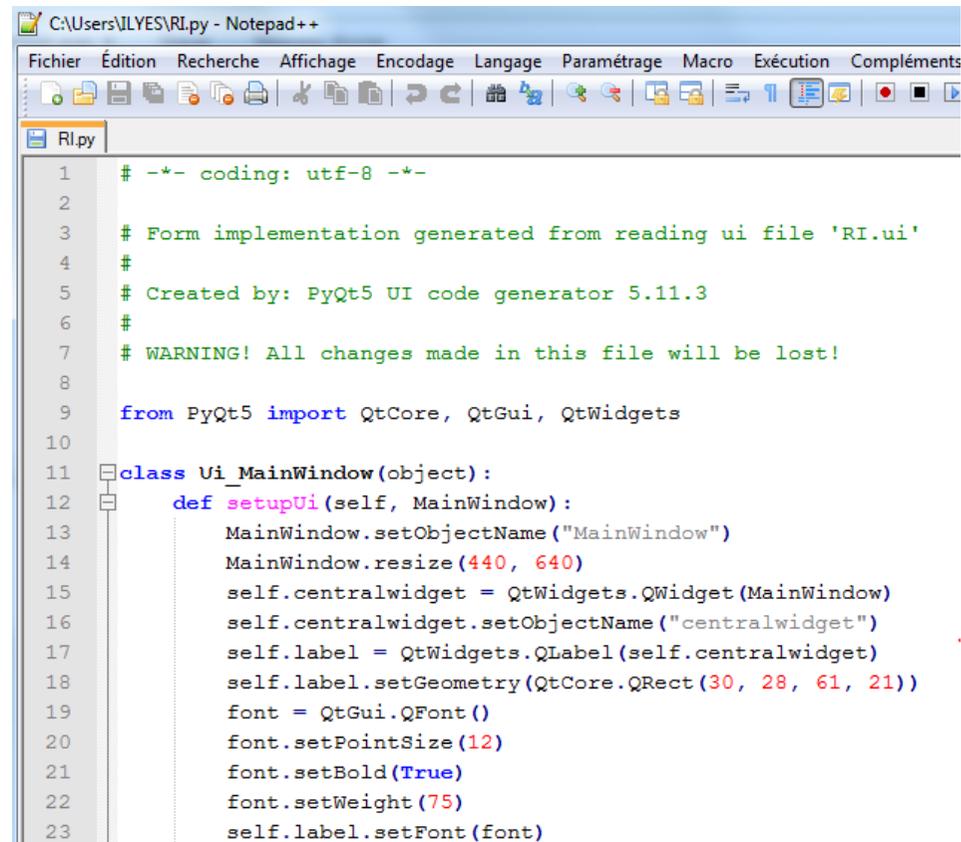


```
Anaconda Prompt

(base) C:\Users\ILYES>Designer

(base) C:\Users\ILYES>pyuic5 -x RI.ui -o RI.py

(base) C:\Users\ILYES>
```



```
C:\Users\ILYES\RI.py - Notepad++
Fichier  Édition  Recherche  Affichage  Encodage  Langage  Paramétrage  Macro  Exécution  Compléments

RI.py
1  # -*- coding: utf-8 -*-
2
3  # Form implementation generated from reading ui file 'RI.ui'
4  #
5  # Created by: PyQt5 UI code generator 5.11.3
6  #
7  # WARNING! All changes made in this file will be lost!
8
9  from PyQt5 import QtCore, QtGui, QtWidgets
10
11 class Ui_MainWindow(object):
12     def setupUi(self, MainWindow):
13         MainWindow.setObjectName("MainWindow")
14         MainWindow.resize(440, 640)
15         self.centralwidget = QtWidgets.QWidget(MainWindow)
16         self.centralwidget.setObjectName("centralwidget")
17         self.label = QtWidgets.QLabel(self.centralwidget)
18         self.label.setGeometry(QtCore.QRect(30, 28, 61, 21))
19         font = QtGui.QFont()
20         font.setPointSize(12)
21         font.setBold(True)
22         font.setWeight(75)
23         self.label.setFont(font)
```

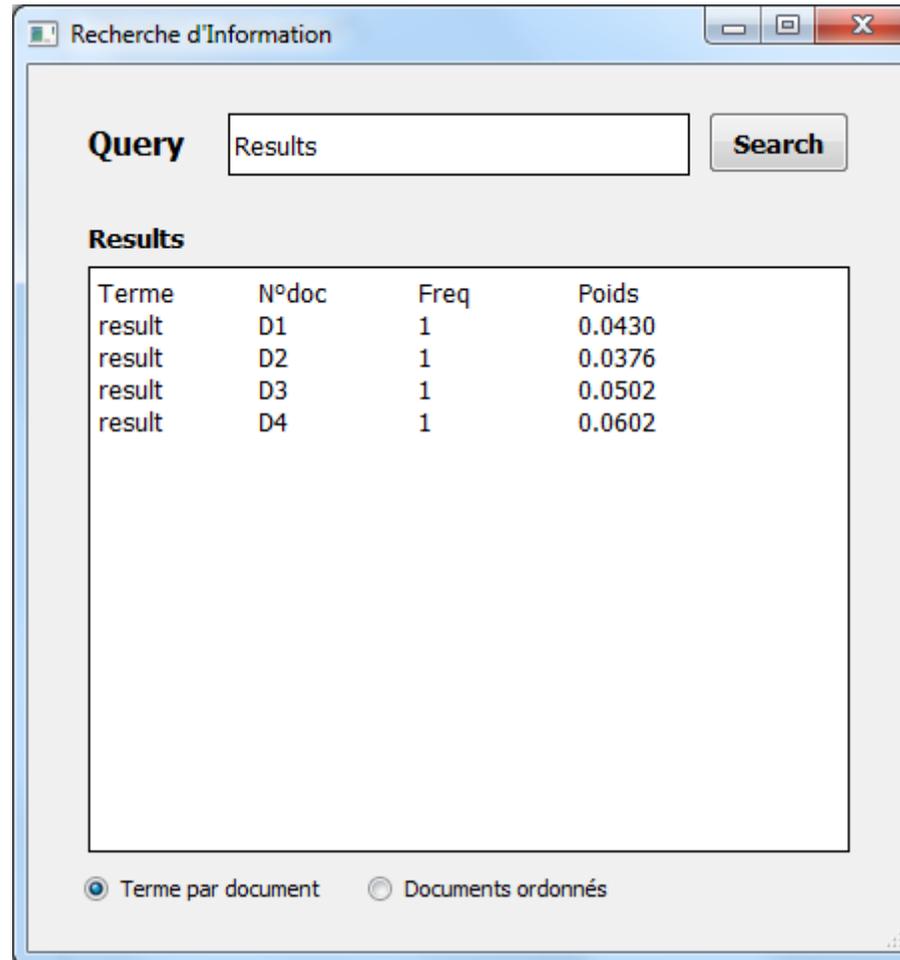
## Appariement :

### Entrée :

Un terme

### Sortie :

Pour chaque document de la collection, en retourne la fréquence et le poids du terme



```
self.pushButton.clicked.connect(self.search)
...
def search(self):
...
```

```
self.lineEdit.text()
```

```
self.textEdit.setText("")
...
self.textEdit.append("")
...
```

```
self.radioButton.setChecked(True)
...
self.radioButton.isChecked() == True
...
```

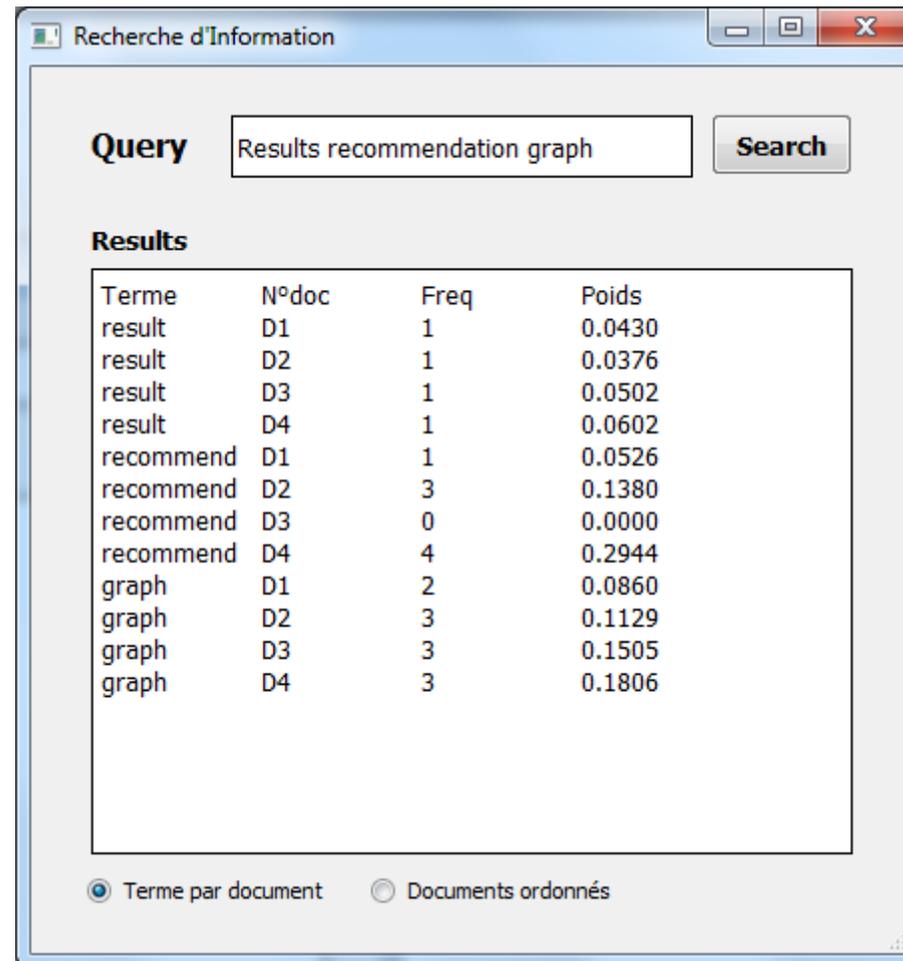
## Appariement :

### Entrée :

Un ensemble de termes

### Sortie :

Pour chaque document de la collection, en retourne la fréquence et le poids de chaque terme de la requête



The screenshot shows a window titled 'Recherche d'Information'. It has a search bar containing the text 'Results recommendation graph' and a 'Search' button. Below the search bar, the results are displayed in a table with four columns: 'Terme', 'N°doc', 'Freq', and 'Poids'. The results are sorted by term, and for each term, the documents are listed with their frequency and weight. At the bottom of the window, there are two radio buttons: 'Terme par document' (which is selected) and 'Documents ordonnés'.

Terme	N°doc	Freq	Poids
result	D1	1	0.0430
result	D2	1	0.0376
result	D3	1	0.0502
result	D4	1	0.0602
recommend	D1	1	0.0526
recommend	D2	3	0.1380
recommend	D3	0	0.0000
recommend	D4	4	0.2944
graph	D1	2	0.0860
graph	D2	3	0.1129
graph	D3	3	0.1505
graph	D4	3	0.1806

## Appariement :

### Entrée :

Un ensemble de termes

### Sortie :

En retourne une liste de documents ordonnés selon leurs degrés de pertinences. Le degré de pertinence d'un document correspond à la somme des poids des termes de ce document appartenant à la requête

