

IRMAR

December 22, 2017

1 Présentation et illustration de l'écosystème Python pour la data science

Francis Wolinski

@fran6wol

Yotta Conseil - Paris 1

1.1 Journée Python et Data Science

IRMAR

Rennes - 19 décembre 2017

1.2 Présentation du langage Python

- Programmation par objets permettant un style de programmation fonctionnelle
- Syntaxe élégante et concise
- Large bibliothèque de modules et d'extensions
- Interprété et disposant d'un mode de programmation interactive
- Portable sur Linux, Mac OS X et Windows...
- Open source et supporté par la Python Software Foundation

```
In [1]: # Hello World!
```

```
def hello_world():  
    print("Hello World!")
```

```
hello_world()
```

Hello World!

```
In [2]: # list comprehension
```

```
[i * i for i in range(10)]
```

```
Out[2]: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
In [4]: # set comprehension
```

```
{i * i for i in range(-10, 11) if i % 2 == 1}
```

```
Out[4]: {1, 9, 25, 49, 81}
```

```
In [5]: # range & slicing
```

```
a = range(100)
b = a[0:100:2]
print(list(b))
```

```
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42,
```

```
In [6]: b[10:20:3]
```

```
Out[6]: range(20, 40, 6)
```

1.3 Les 3 étapes du langage Python

- Les années 1990 : Python 1
 - Langage de script
 - Une alternative à bash
- Les années 2000 : Python 2
 - Calcul scientifique
 - Une alternative à MATLAB
- Les années 2010 : Python 3
 - Data science
 - Une alternative à R

Source : Jake VanderPlas (University of Washington)

2 Le succès actuel de Python

Python has continued its upward trajectory from last year and jumped two places to the No. 1 slot, though the top four—Python, C, Java, and C++—all remain very close in popularity.

Source IEEE : <https://spectrum.ieee.org/computing/software/the-2017-top-programming-languages>

2.1 Python : un langage universel

2.2 Python et la data science

Python, whose usage has been growing faster than R for the last several years, has finally caught up with R, and (barely) overtook it, with 52.6% respondents using it v. 52.1% for R.

Source : KDNuggets 2017 Poll <http://www.kdnuggets.com/2017/05/poll-analytics-data-science-machine-learning-software-leaders.html>























<div> <div>Web</div> <div>Mobile</div> <div>Enterprise</div> <div>Embedded</div> </div>			
Language Rank	Types	Spectrum Ranking	Custom Ranking
1. Python	 	100.0	100.0
2. C	  	99.7	98.1
3. Java	  	99.4	98.0
4. C++	  	97.2	95.9
5. C#	  	88.6	87.7
6. R		88.1	86.5
7. JavaScript	 	85.5	82.5
8. PHP		81.4	81.8
9. Go	 	76.1	74.2
10. Swift	 	75.3	71.5

image2.png



image1.png

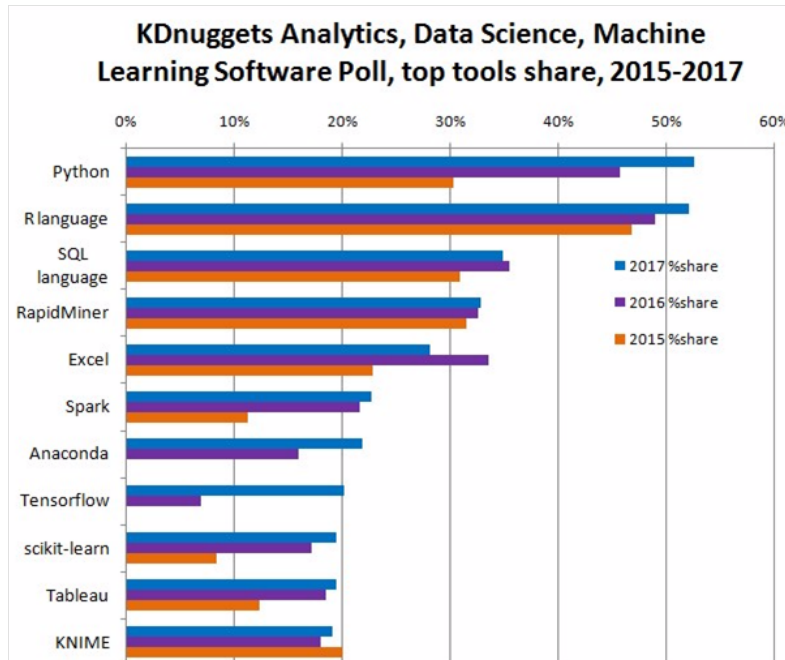


image3.png

3 Ecosystème Python pour la Data Science

3.1 Succès des librairies Python

Source : David Robison – Why is Python Growing So Quickly ?
<https://stackoverflow.blog/2017/09/14/python-growing-quickly/>

```
In [ ]: # launch spyder
```

```
%run C:/Test/Anaconda3/Scripts/spyder-script.py
```

```
In [7]: # fonctions de Bessel
```

```
from numpy import linspace, pi
from scipy.special import jn
from matplotlib.pyplot import plot
```

```
%matplotlib inline
```

```
x = linspace(0, 4*pi)
for i in range(5, -1, -1):
    plot(x, jn(i, x))
```

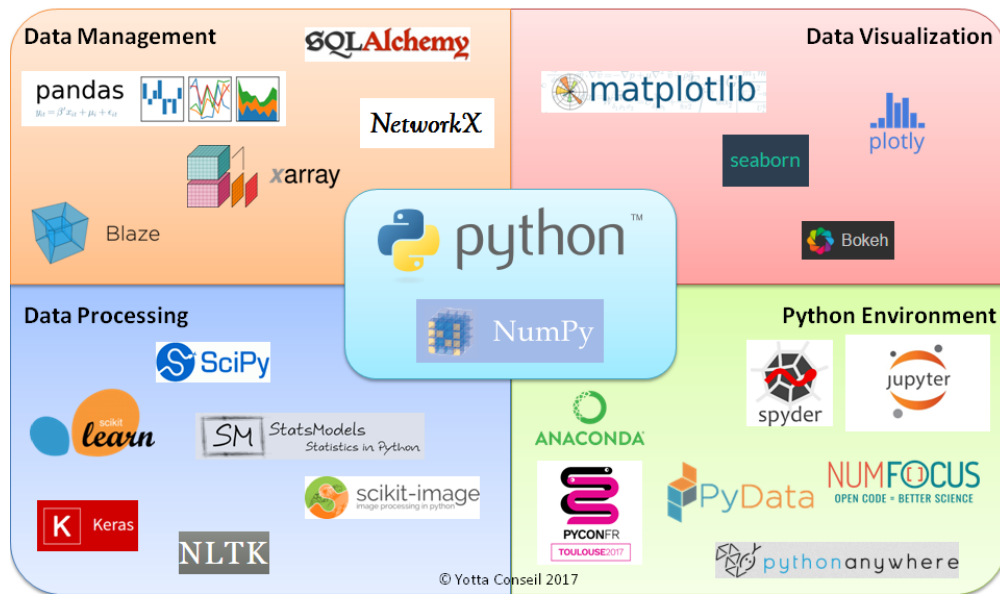


image5.png

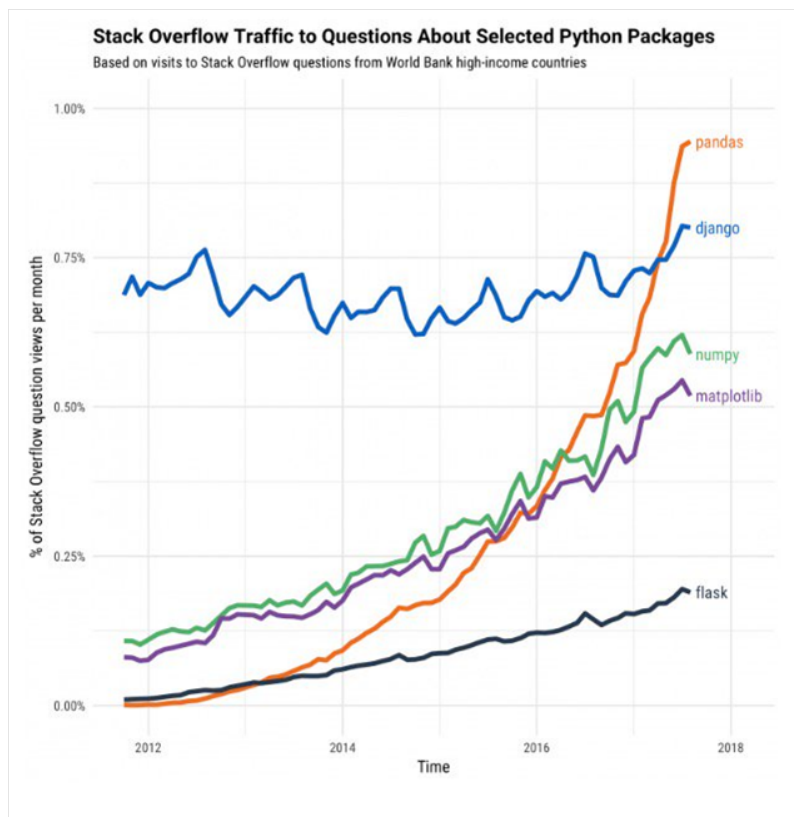
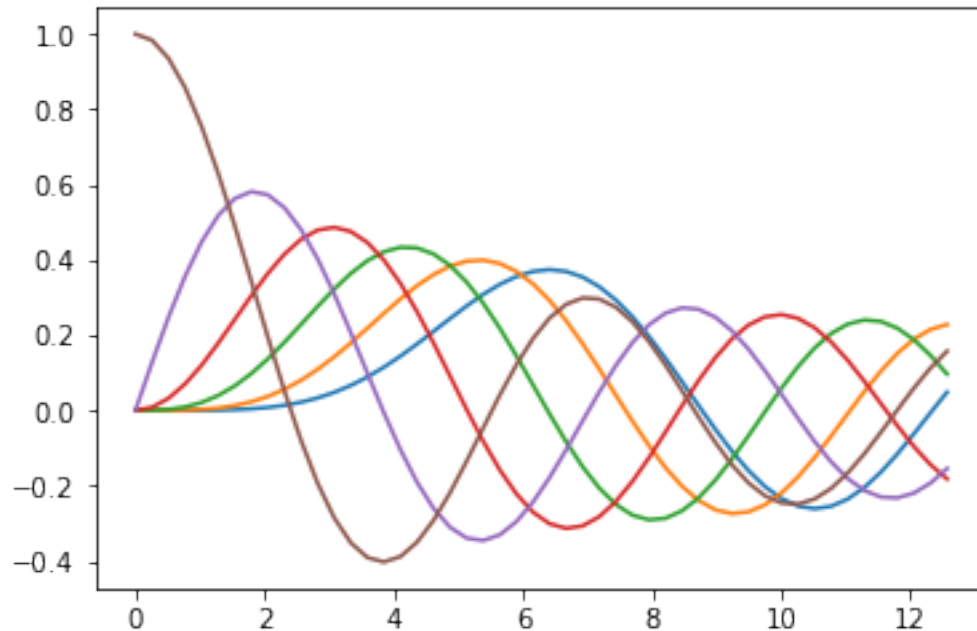


image6.png



$$J_\nu(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{n! \Gamma(n+1+\nu)} \left(\frac{x}{2}\right)^{2n+\nu}$$

4 La révolution Jupyter

```
In [8]: # Interface multimedia
        from IPython.display import IFram
        IFram("http://calcul.math.cnrs.fr/spip.php?article287", width=800, height=
```

```
Out[8]: <IPython.lib.display.IFrame at 0x4de6470>
```

4.1 NumPy

Librairie destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux

Caractéristiques principales - Structure de données ndarray (N-dimentional array) typée dtype (data-type) - Calculs vectoriel et matriciel - Algèbre linéaire, nombres aléatoires, transformations de Fourier... - Outils pour intégrer du C++ et du Fortran

NumPy est optimisé par rapport à Python : ~ x100 pour des boucles numériques

```
In [9]: # Performances de Python vs NumPy

        # boucle Python
        %timeit [i * i for i in range(10000000)]
```

```
1 loop, best of 3: 943 ms per loop
```

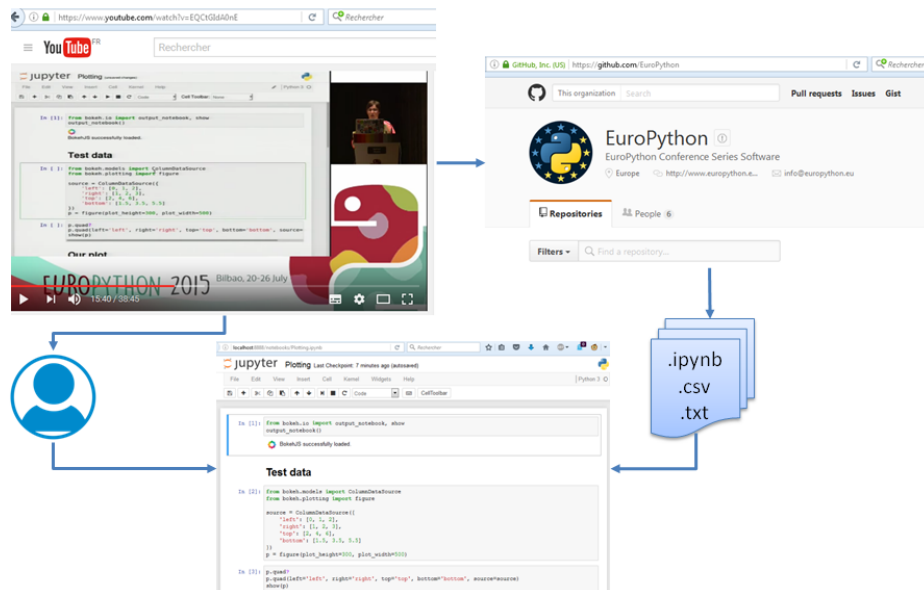


image7.png

```
In [10]: # ufunc NumPy
import numpy as np
%timeit np.arange(1e7) ** 2
```

10 loops, best of 3: 63.3 ms per loop

4.2 Matplotlib

Librairie destinée à tracer et visualiser des données sous formes de graphiques 2D

Caractéristiques principales

- Graphes de fonctions
- Histogrammes
- Diagrammes de densité
- Graphiques en barres et secteurs
- Nuages de points

Nouvelles librairies

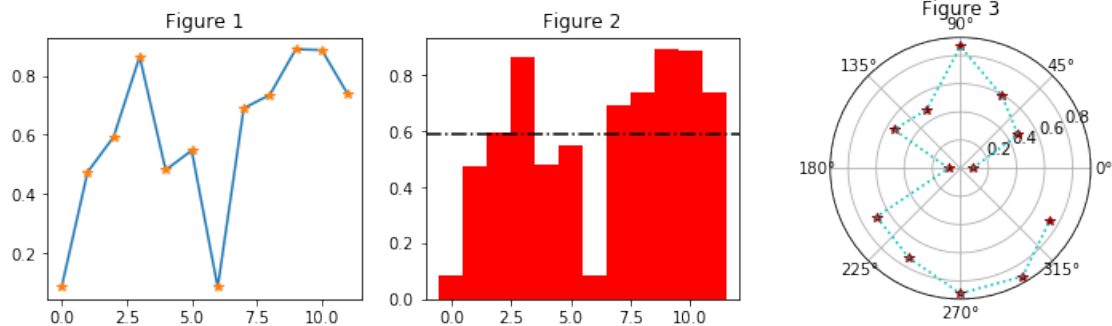
- seaborn: statistical data visualization
- bokeh: statistical and novel interactive HTML plots for Python

```
In [11]: import matplotlib.pyplot as plt
x = np.random.random(12)
```

```

fig = plt.figure(figsize=(12, 3))
ax1 = fig.add_subplot(131)
ax1.set_title("Figure 1")
ax1.plot(x)
ax1.plot(x, '*')
ax2 = fig.add_subplot(132)
ax2.set_title("Figure 2")
ax2.bar(np.arange(12), x, 1, color='r')
ax2.axhline(x.mean(), color='k', linestyle='-.')
ax3 = fig.add_subplot(133, projection='polar')
ax3.set_title("Figure 3")
r = np.linspace(0, (2 - 1 / 6) * np.pi, 12)
ax3.plot(r, x, '*', color='darkred')
ax3.plot(r, x, 'c:');

```



4.3 SciPy

Librairie visant à unifier et fédérer un ensemble de bibliothèques Python à usage scientifique

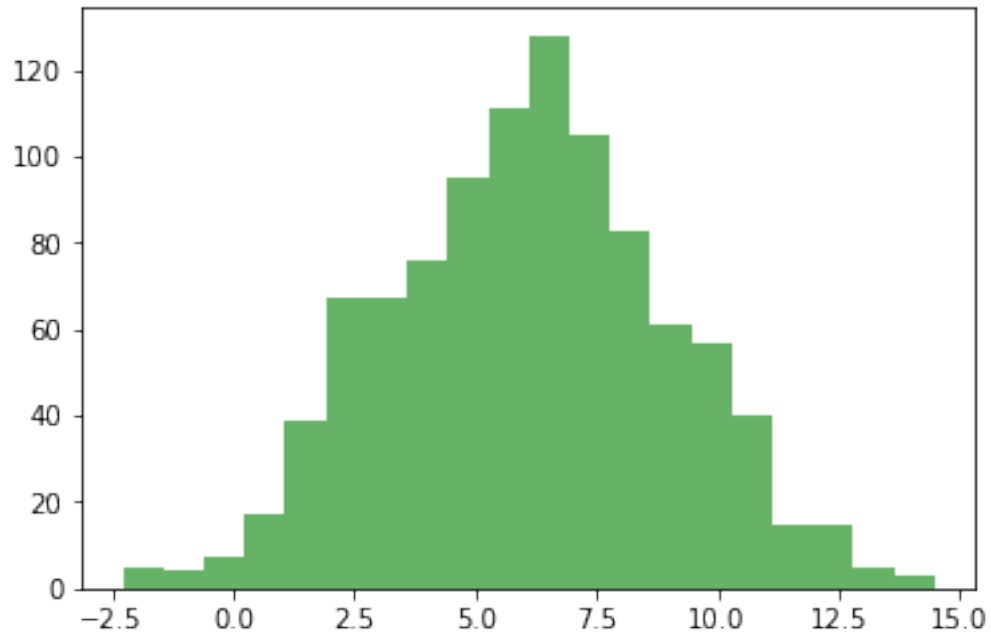
Caractéristiques principales - Statistiques - Lois discrètes ou continues - Statistiques descriptives - Tests statistiques - Optimisation - Algèbre linéaire - Matrices creuses - Traitement du signal et d'images

```

In [12]: # random normal distribution
mu = np.random.randint(10)
std = np.random.randint(1, 4)
print("mu={:.2f}, std={:.2f}".format(mu, std))
a = np.random.normal(mu, std, 1000)
plt.hist(a, bins=20, color='g', alpha=0.6);

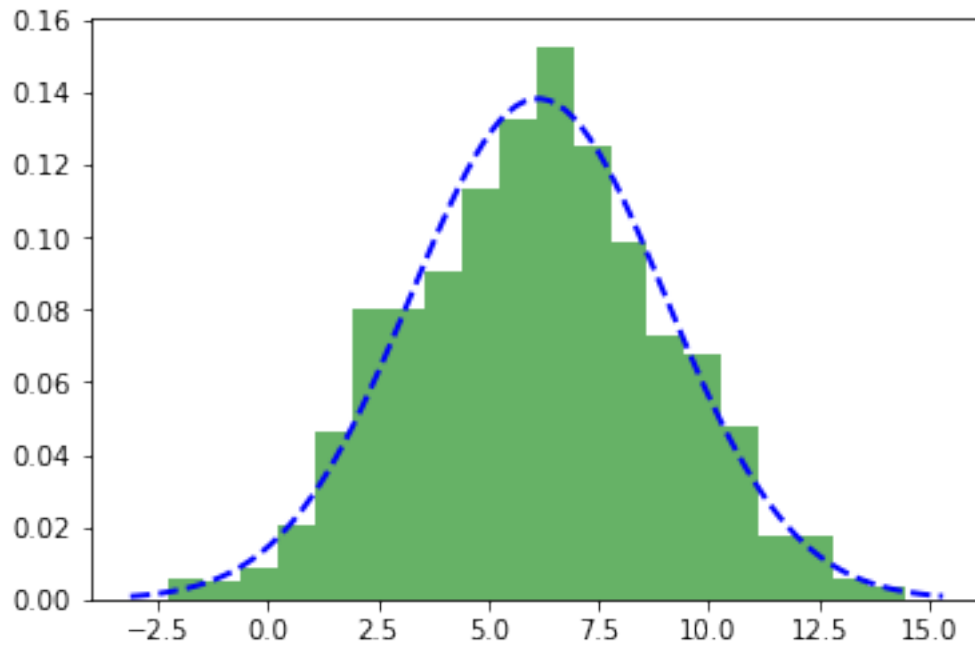
```

mu=6.00, std=3.00



```
In [13]: # estimation of a normal distribution
from scipy.stats import norm
mu, std = norm.fit(a)
print("mu={0:.2f}, std={1:.2f}".format(mu, std))
plt.hist(a, bins=20, normed=True, color='g', alpha=0.6)
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
y = norm.pdf(x, mu, std)
plt.plot(x, y, 'b--', linewidth=2);

mu=6.13, std=2.89
```



4.4 Pandas

Librairie fournissant des structures de données et les outils d'analyse associés performants et faciles à utiliser

Caractéristiques principales

- Structure 1D : Series
- Structure 2D : DataFrame
- Intégrée avec NumPy, SciPy et matplotlib...

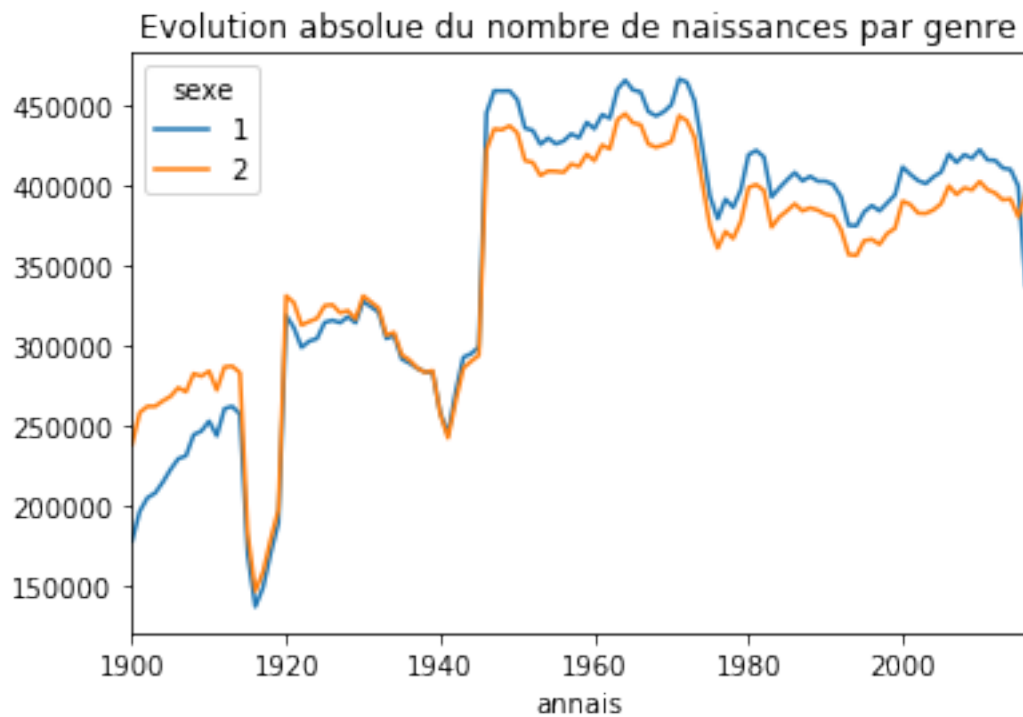
```
In [14]: import pandas as pnd
          df = pnd.read_csv("c:/Test/mydata/irmar/nat2015.txt",
                           sep="\t",
                           encoding='latin-1')

          df.head(10)
```

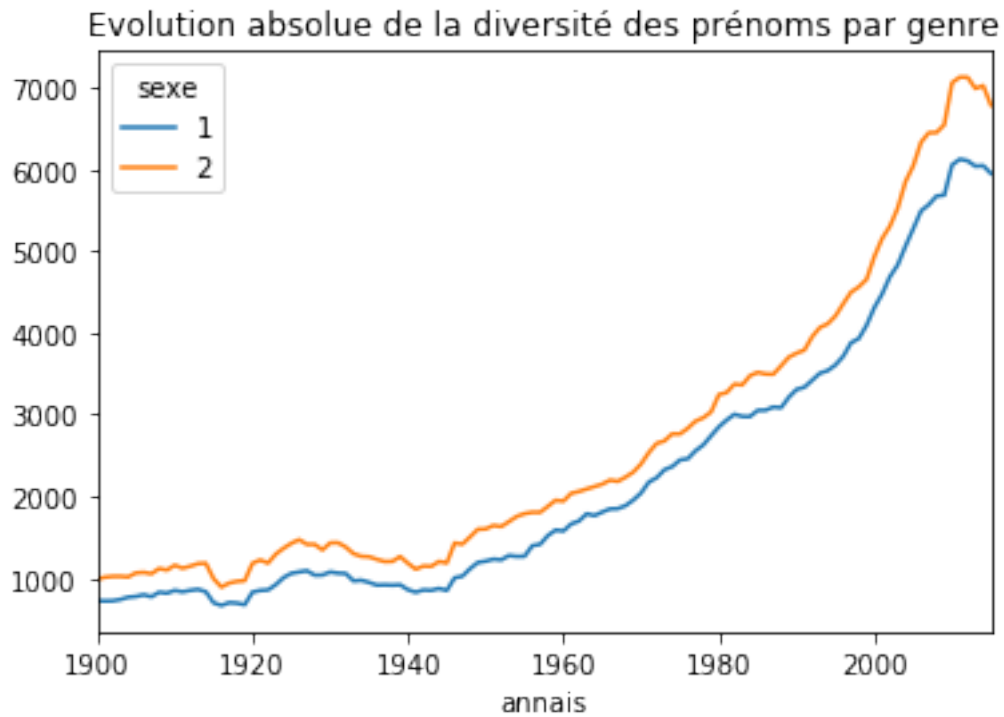
```
Out [14]:
```

	sexe	preusuel	annais	nombre
0	1	A	1980	3.0
1	1	A	1998	3.0
2	1	A	XXXX	21.0
3	1	AADEL	1976	5.0
4	1	AADEL	1978	3.0
5	1	AADEL	1980	3.0
6	1	AADEL	1981	5.0
7	1	AADEL	1982	4.0
8	1	AADEL	1983	3.0
9	1	AADEL	1987	5.0

```
In [15]: tab = df.pivot_table(index="annais", columns="sexe", values="nombre", aggfunc="sum")
tab.plot(title="Evolution absolue du nombre de naissances par genre");
```

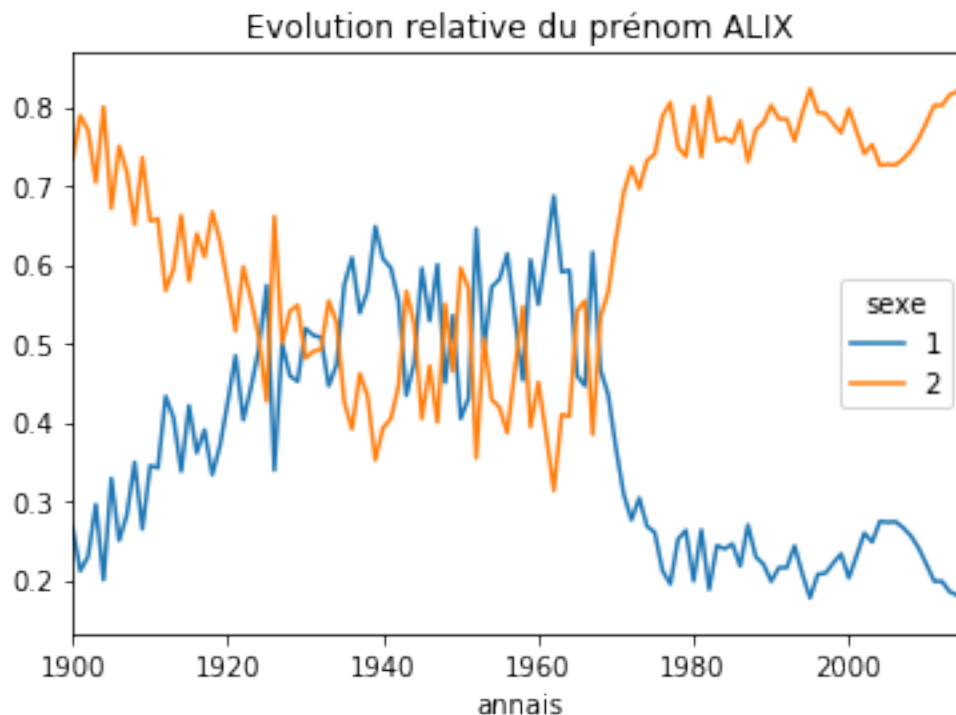


```
In [16]: df = df.loc[df["annais"] != "XXXX"]
tab = df.pivot_table(index="annais", columns="sexe", values="preusuel", aggfunc="sum")
tab.plot(title="Evolution absolue de la diversité des prénoms par genre");
```



```
In [19]: def evol_prenom(prenom):
    sel = df.loc[df['preusuel'] == prenom]
    evol = sel.pivot_table(index='annais',
                           columns='sexe',
                           values='nombre',
                           aggfunc='sum')
    tab = evol.div(evol.sum(axis=1), axis=0)
    tab.plot(title="Evolution relative du prénom {}".format(prenom));

evol_prenom('ALIX')
```



4.5 scikit-learn

Librairie de machine learning et de data mining

Caractéristiques principales

- Classification (SVM, plus proches voisins, random forest...)
- Régression (linéaire, logistique...)
- Clustering (partitionnement en k-means, ...)
- Réduction de dimension (analyse en composantes principales, ...)
- Sélection de modèle (validations croisées...)
- Intégrée avec NumPy, SciPy et matplotlib
- Keras (réseaux de neurones) fournit une API à la scikit-learn

```
In [20]: from sklearn.model_selection import train_test_split
```

```
X = np.random.random((2, 1000))
y = X[1] > np.sin(np.pi * X[0])
X_train, X_test, y_train, y_test = train_test_split(X.T, y, test_size=0.3)

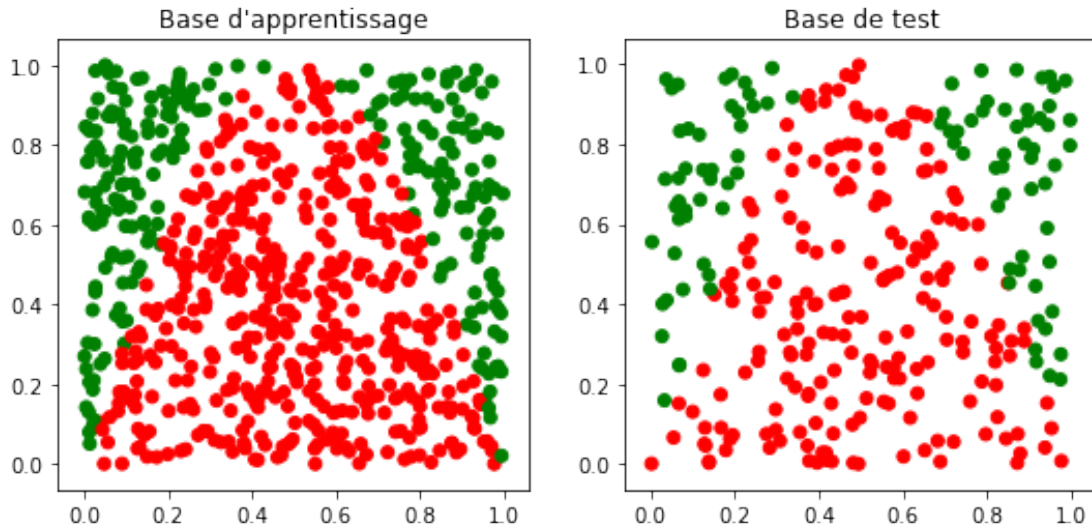
c_train = np.where(y_train, ['g'] * len(y_train), ['r'] * len(y_train))
c_test = np.where(y_test, ['g'] * len(y_test), ['r'] * len(y_test))

fig = plt.figure(figsize=(9, 4))
ax1 = fig.add_subplot(121)
```

```

ax1.set_title("Base d'apprentissage")
ax1.scatter(X_train.T[0], X_train.T[1], color=c_train)
ax2 = fig.add_subplot(122)
ax2.set_title("Base de test")
ax2.scatter(X_test.T[0], X_test.T[1], color=c_test);

```



```

In [21]: from sklearn.linear_model import LogisticRegression
         from sklearn.svm import SVC
         from sklearn.neural_network import MLPClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.ensemble import RandomForestClassifier

         from sklearn.metrics import accuracy_score

```

```

In [23]: algo = RandomForestClassifier()

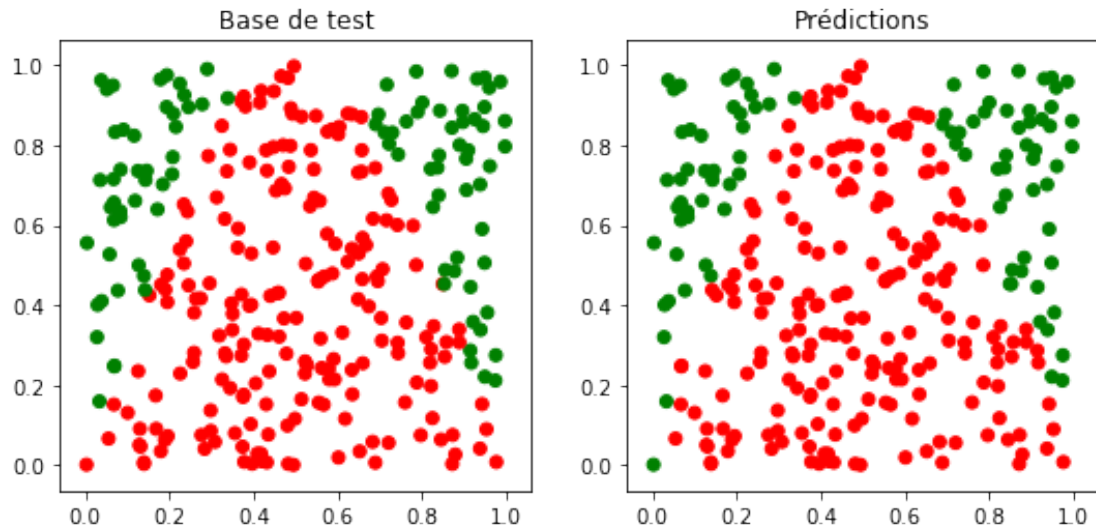
         algo.fit(X_train, y_train)
         y_pred = algo.predict(X_test)
         c_pred = np.where(y_pred, ['g'] * len(y_pred), ['r'] * len(y_pred))

         acc_test = accuracy_score(y_test, y_pred)
         print("Accuracy: {:.2f}%".format(acc_test))

         fig = plt.figure(figsize=(9, 4))
         ax1 = fig.add_subplot(121)
         ax1.set_title("Base de test")
         ax1.scatter(X_test.T[0], X_test.T[1], color=c_test)
         ax2 = fig.add_subplot(122)
         ax2.set_title("Prédictions")
         ax2.scatter(X_test.T[0], X_test.T[1], color=c_pred);

```

Accuracy: 0.98%



4.6 Conclusion et Références

- Travis Oliphant – Guide to NumPy (2015)
- Jake VanderPlas - Python Data Science Handbook (2016)
- Wes McKinney - Python for Data Analysis: Data Wrangling with Pandas, Numpy, and IPython (2017)
- Python <https://www.python.org/>
- SciPy <https://www.scipy.org/>
- NumPy <http://www.numpy.org/>
- Pandas <http://pandas.pydata.org/>
- Matplotlib <https://matplotlib.org/>
- Scikit Learn <http://scikit-learn.org>
- Jupyter <http://jupyter.org/>
- Anaconda <https://www.anaconda.com/>
- NumFocus <https://www.numfocus.org/>

In []: