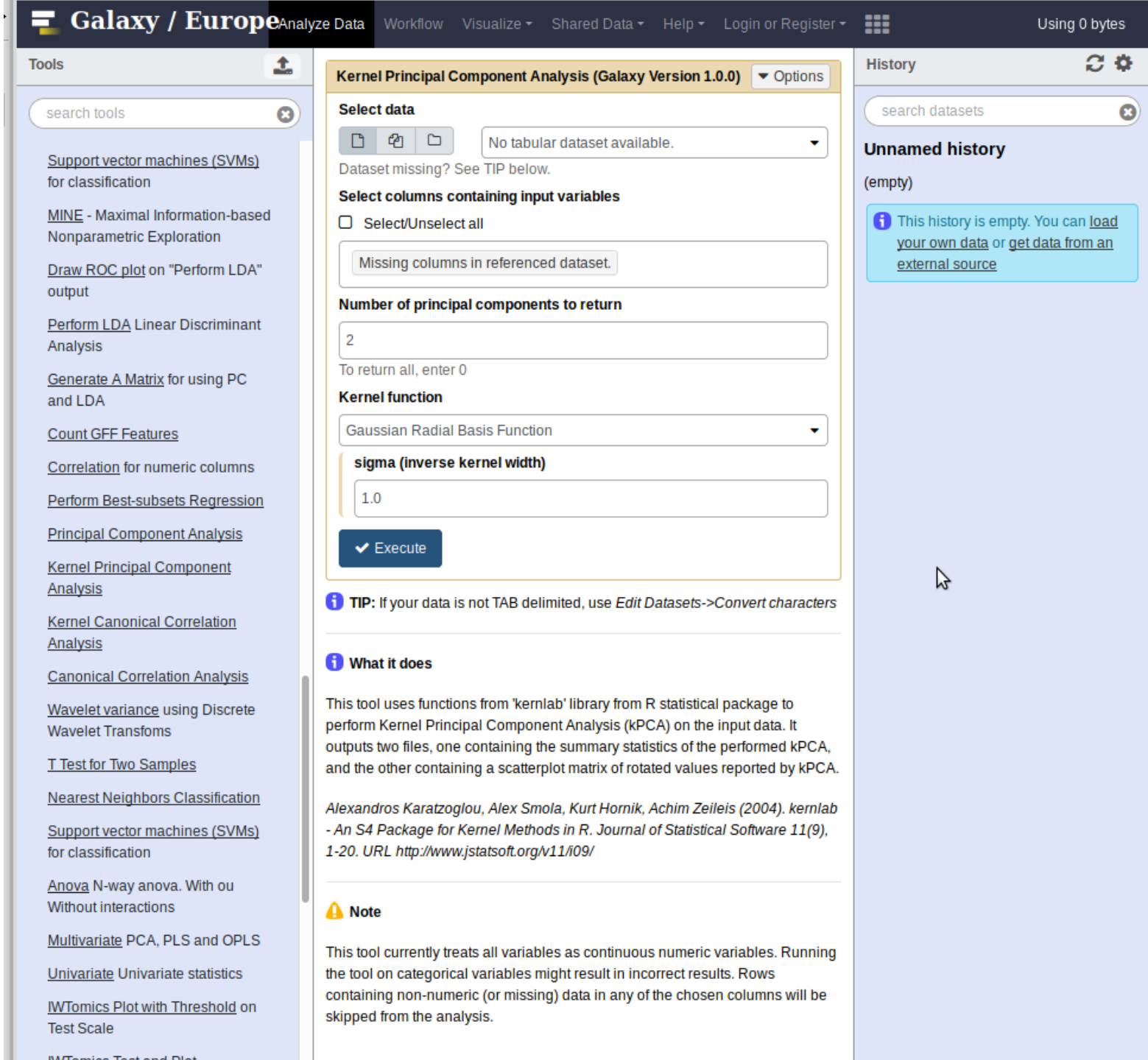
Mise en œuvre de solutions de reproductibilité à l'aide des principaux notebook et d'un système de workflow

Antoine COSSA
Marine FIGAROL
Cécile MOUREAUX
Christelle VIGUIER

Galaxy
PROJECT



Tools

Workflow Canvas | BigData

search tools

Inputs
- Input dataset
- Input dataset collection

FILE AND META TOOLS

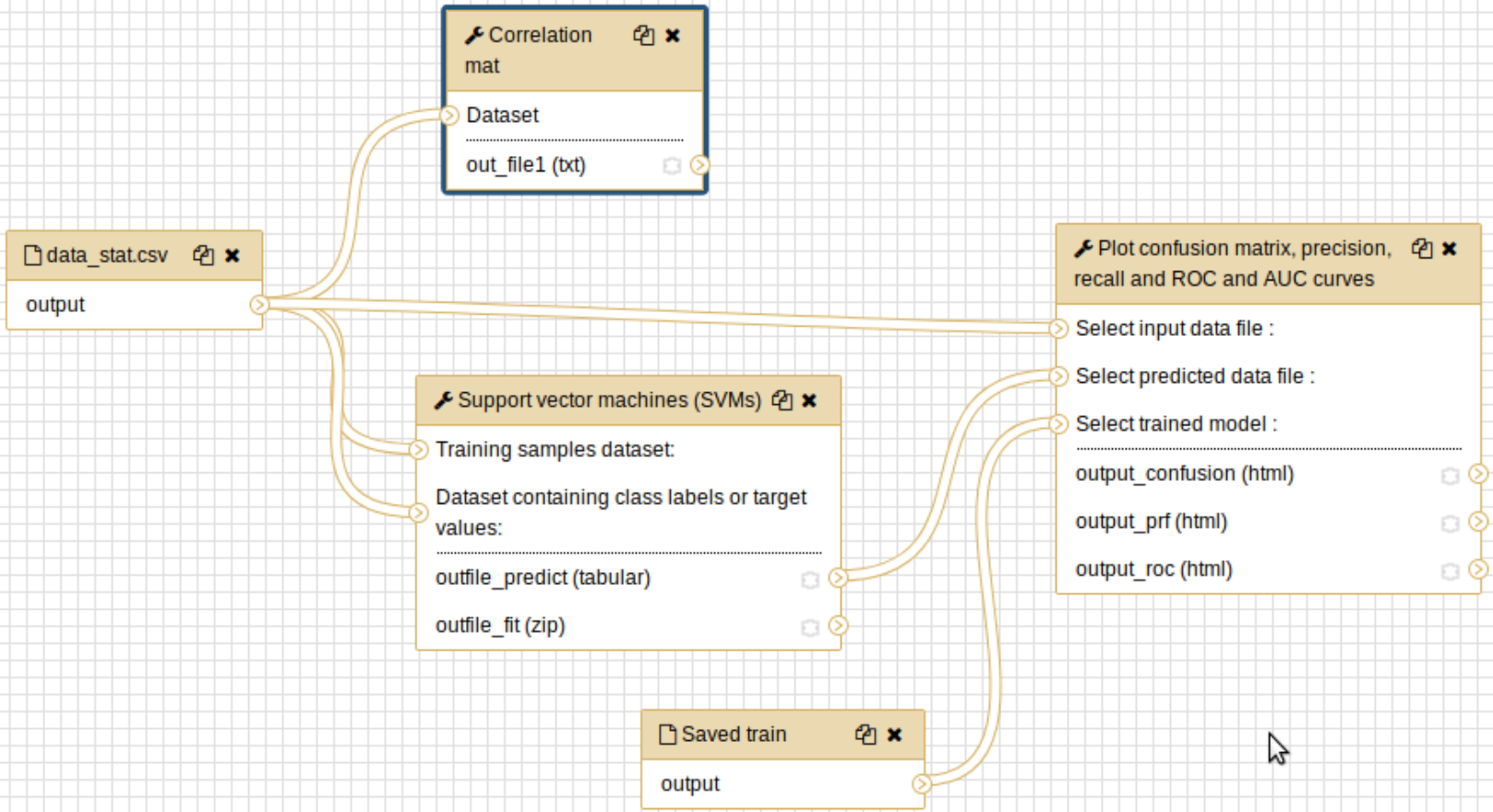
Get Data
Send Data
Convert Formats
Collection Operations

GENERAL TEXT TOOLS

Text Manipulation
Filter and Sort
Join, Subtract and Group

GENOMICS, NGS

Extract Features
BED Tools
Fetch Alignments
Operate on Genomic Intervals
FASTA/FASTQ manipulation
Multiple Alignments
FASTA/FASTQ manipulation
Picard
Quality Control
Assembly
Mapping
Variant Calling
Genome editing



Notebooks

- Interface web + Kernel
- Combine texte formaté / code
- Multi-langage
- Portabilité / Reproductibilité

Installation

- Installation rapide sous Windows et Linux

```
python3 –m pip install –upgrade pip

python3 –m pip install jupyter
```

- Lancement facile et rapide

```
jupyter notebook
```

Python nécessaire pour installer Jupyter Notebook

- Téléchargement : https://www.rstudio.com/products/rstudio/download/

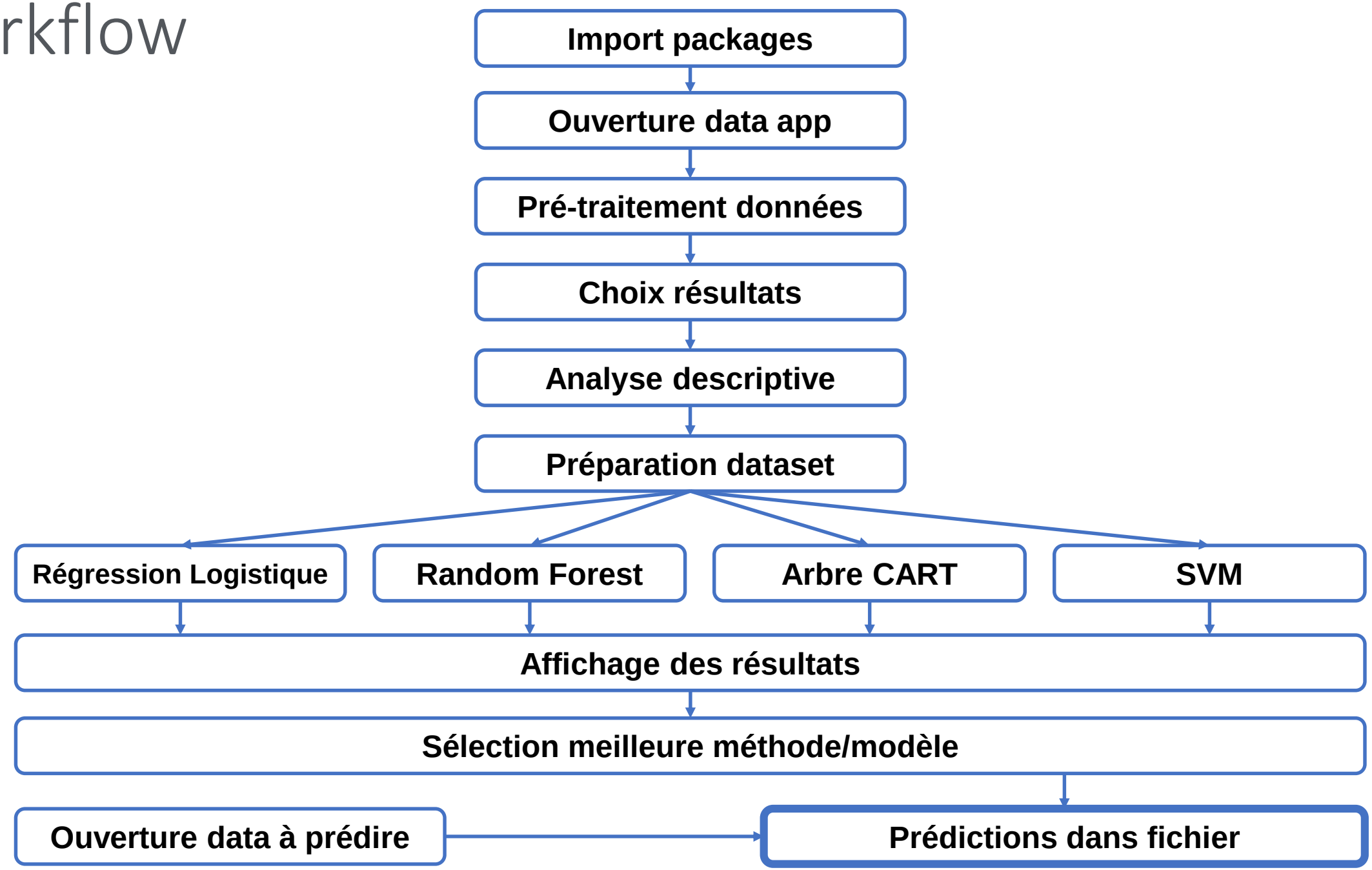
- Plusieurs installations possibles. Assez rapide sous Linux ; très compliquée sous Windows

JAVA, Hadoop, Spark, Git, Scala, Maven nécessaires

- Lancement facile et rapide sous Linux et Windows

```
bin/zeppelin-daemon.sh start
bin\zeppelin.cmd
```

Workflow


Import packages
Ouverture data app
Pré-traitement données
Choix résultats
Analyse descriptive
Préparation dataset
Régression Logistique
Random Forest
Arbre CART
SVM
Affichage des résultats
Sélection meilleure méthode/modèle
Ouverture data à prédire
Prédictions dans fichier

Dataset1 : Cancer du sein

Variable d'intérêt : rechute

Co variables : niveau d'expression de gènes

But : prédire la rechute ou non-rechute des patientes

Rechute	Non Rechute
1	-1

Dataset2 : Spam (test de la reproductibilité)

Variable d'intérêt : type

Co variables : fréquence de certains mots et caractères

But : prédire si il s'agit d'un spam ou non

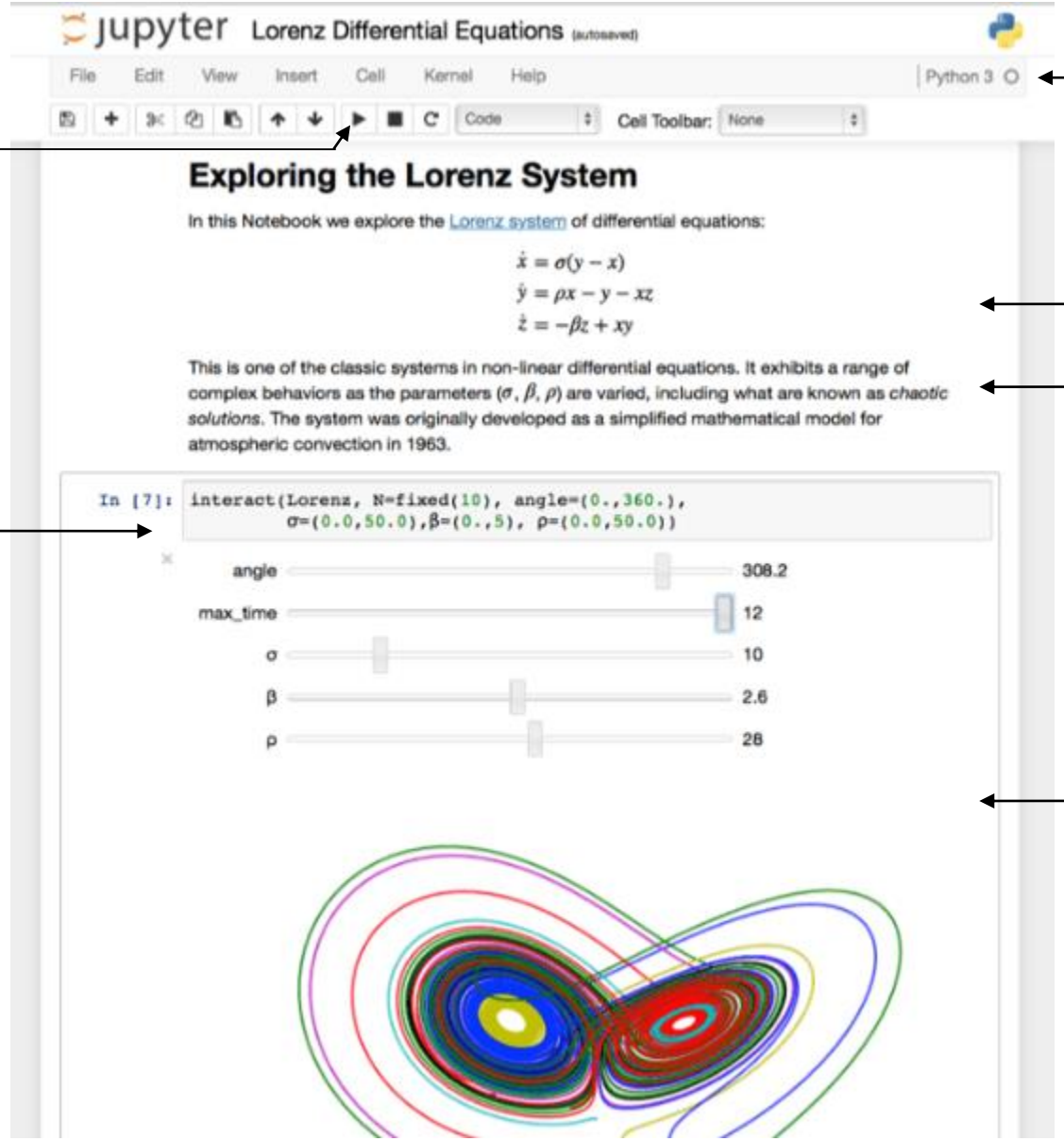
Spam	Non spam
spam	nonspam

Présentation générale

Interface


jupyter
Quit
Logout
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
0
/ Desktop / Projet_Big_Data
Name
Notebook:
Python 3
R
Other:
Text File
Folder
Terminal
BD.ipynb
R_BD.ipynb
Actif
COSSA-MOUREAUX.RMD
donnees_liver.rda
spam.csv
il y a 8 jours
743 kB
train.txt
il y a 19 heures
93.5 kB

Exemple



Exécution de la cellule

Langage utilisé

Texte enrichi

Texte brut

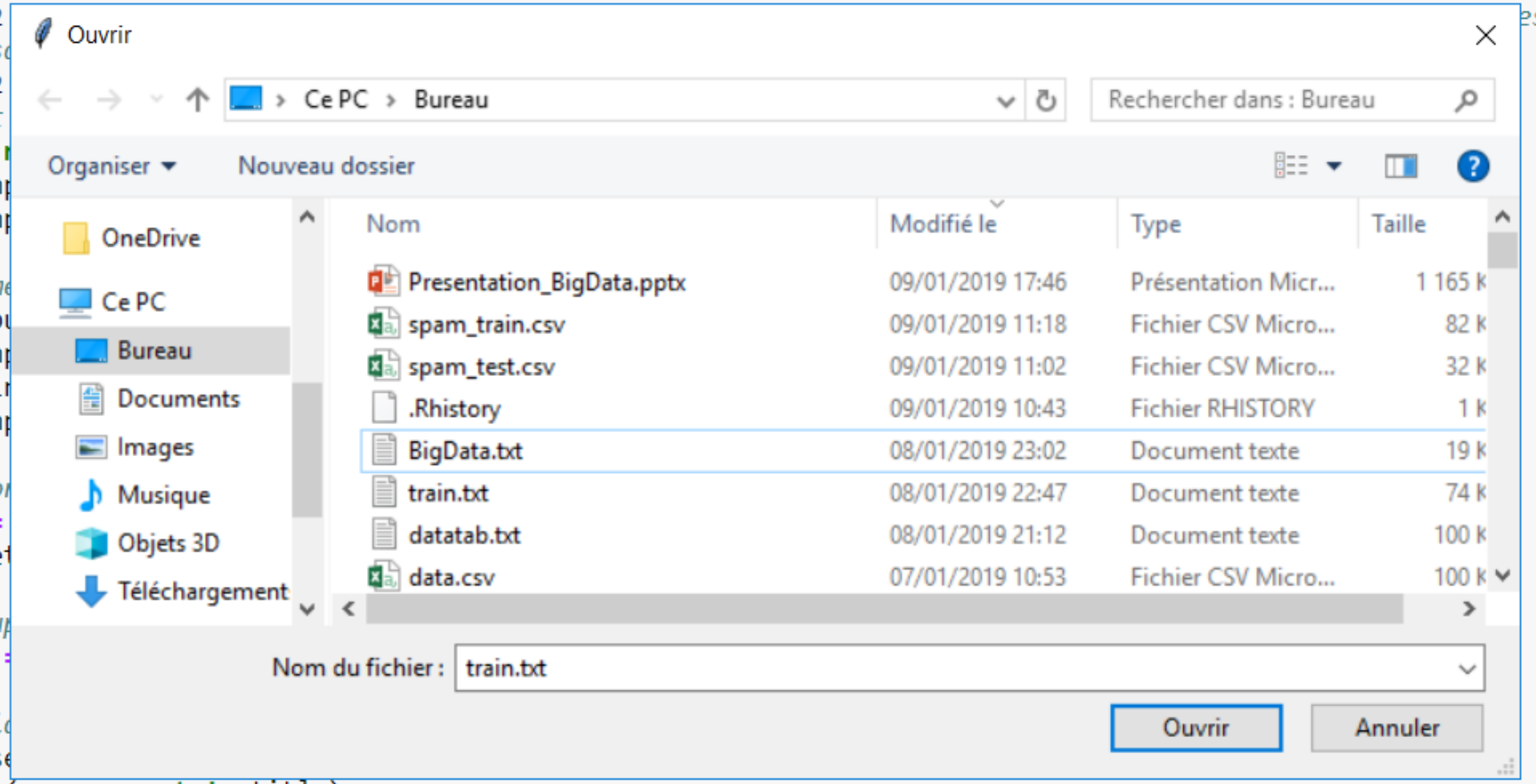
Code

Sortie

Jupyter avec Python

Sélection dataset

```
# Permet à l'utilisateur de choisir un fichier
root = Tk()
root.withdraw()
root.call('wm', 'attributes', '.', '-topmost', True)
input1 = filedialog.askopenfilename()
%gui tk

input2
# Pass
input2
# Tant
while
    inp
    inp

# On me
if inpu
    inp
elif in
    inp

# Impor
file =
dataset

# Récup
title =

# Vari
reponse
while (reponse not in title) :
```



Clé de la reproductibilité : les inputs

```
input2 = input("Indiquez la manière dont sont séparées vos variables (tabulation/espace/;) : ")
# Passage en minuscule si l'utilisateur écrit en majuscule
input2 = input2.lower() ;
# Tant que l'utilisateur n'a pas rentré quelque chose de correct
while not (input2 == 'tabulation' or input2 == 'espace' or input2 == ';') :
    input2 = input("Veuillez entrer tabulation, espace ou ; : ")
    input2 = input2.lower()
```

Indiquez la manière dont sont séparées vos variables (tabulation/espace/;) :

```
reponse = input("Indiquez le nom de la variable d'intérêt : ")
while (reponse not in title) :
    reponse = input("Cette variable n'existe pas ! Veuillez vérifier son nom et essayer à nouveau : ")
```

Indiquez le nom de la variable d'intérêt :

Statistiques descriptives

Entrée [4]:
```
dataset.head()
```

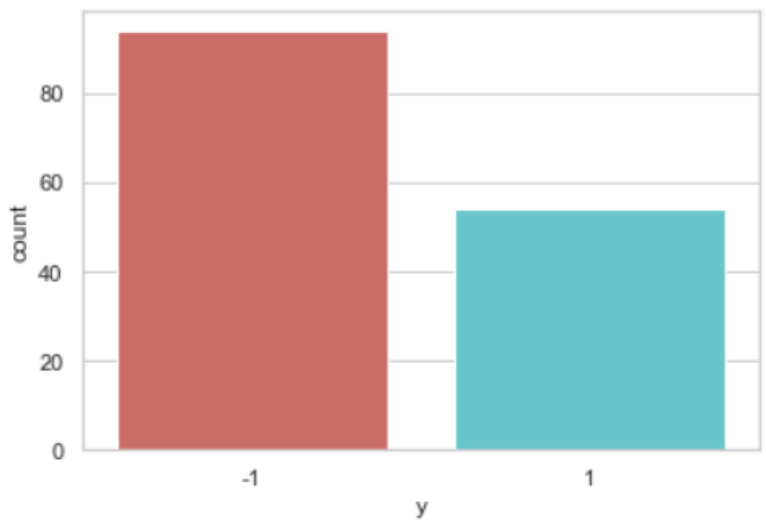
Out[4]:

	x10006_at	x10007_at	x100129361_at	x100130449_at	x100130633_at
0	8.44	7.84	9.23	2.20	4.93
1	7.93	8.47	7.77	2.48	6.70
2	7.53	7.49	9.49	5.66	5.99
3	7.72	8.38	8.03	5.22	6.95
4	7.89	7.75	6.99	2.20	7.06

5 rows × 101 columns

Répartition de la variable d'intérêt

Entrée [8]:
```
sns.countplot(x = reponse, data = dataset, palette = 'hls')
plt.show()
```



Dimension de votre dataset

```
print(dataset.shape)
```

```
(148, 101)
```

Visualisation du résumé de la variable d'intérêt

```
y.describe()
classe = list(set(y))
nb_classe =  len(list(set(y)))

print("Différentes classes de la variable d'intérêt : ", classe)
print("Nombre de classes : ", nb_classe)
```

```
Différentes classes de la variable d'intérêt :  [1, -1]
Nombre de classes :  2
```

Analyses statistiques

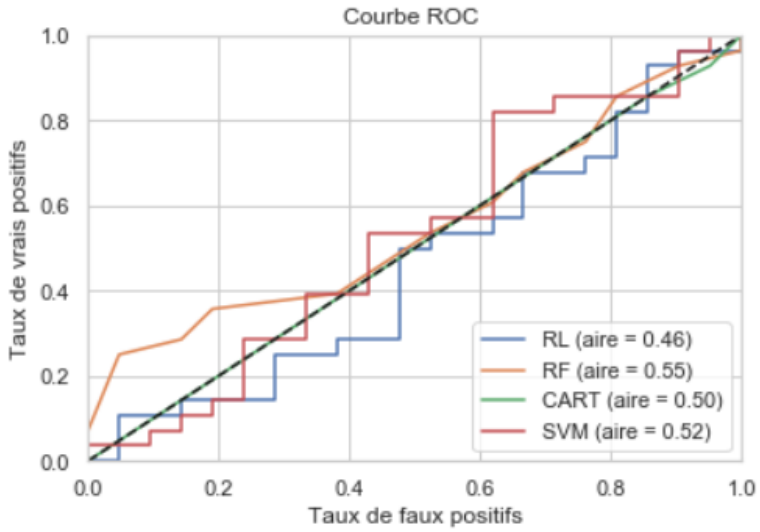
Random Forest

Entrée [14]:

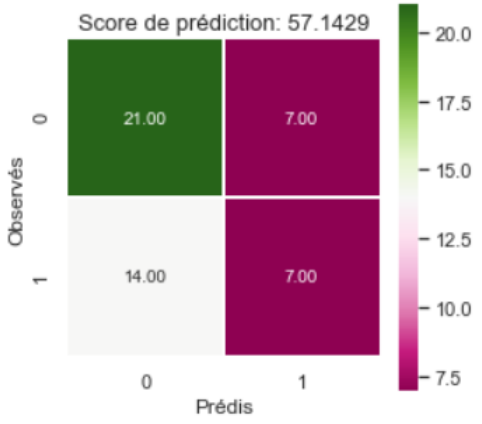
```
rf_model = RandomForestClassifier()

# Etablissement des différents paramètres à tester
parameter_grid = {'n_estimators': [10, 25, 50, 100],
                  'criterion': ['gini', 'entropy'],
                  'max_features': [1, 2, 3, 4]}

# Cross Validation (on fait tourner 10 fois le modele sur differents découpage)
cross_validation = StratifiedKFold(n_splits=10)

# Sélection des meilleurs paramètres
grid_search = GridSearchCV(rf_model,
                           param_grid=parameter_grid,
                           cv=cross_validation)

grid_search.fit(XTrain, yTrain_)

# Visualisation des meilleurs paramètres
print('Best parameters: {}'.format(grid_search.best_params_))

# Stockage des meilleurs paramètres
nestim_best = grid_search.best_params_['n_estimators']
criterion_best = grid_search.best_params_['criterion']
max_features_best = grid_search.best_params_['max_features']
```

Affichage des résultats

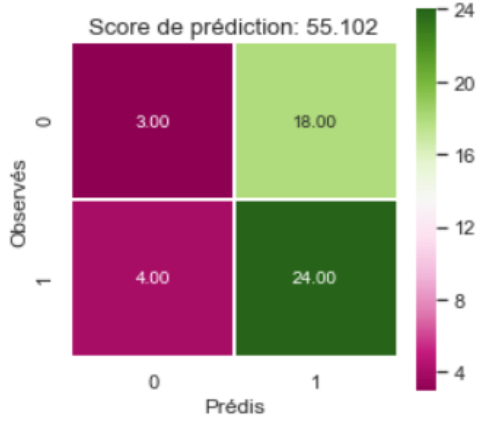
```
Précision de la Régression Logistique :  57.1429 %
Précision du Random Forest :  57.1429 %
Précision de l'arbre CART :  55.102 %
Précision de SVM :  57.1429 %
```



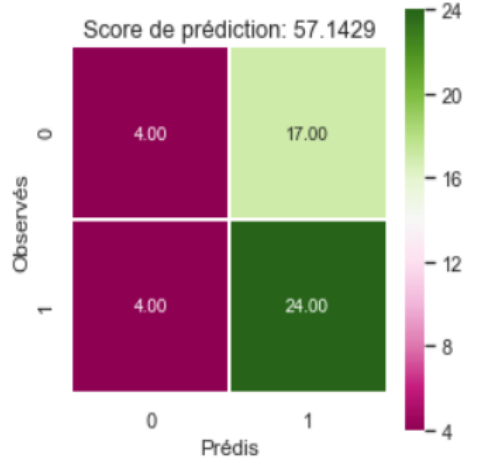
Matrice de confusion de la Régression Logistique:



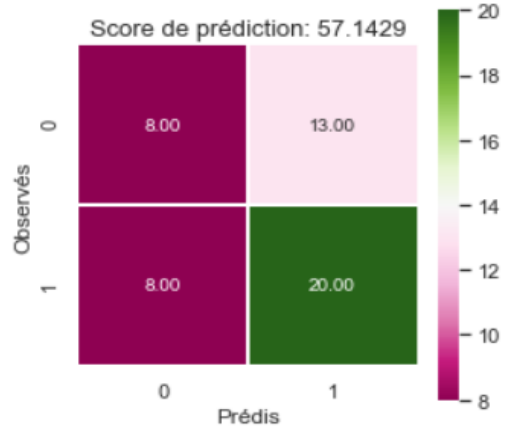
Matrice de confusion de CART:



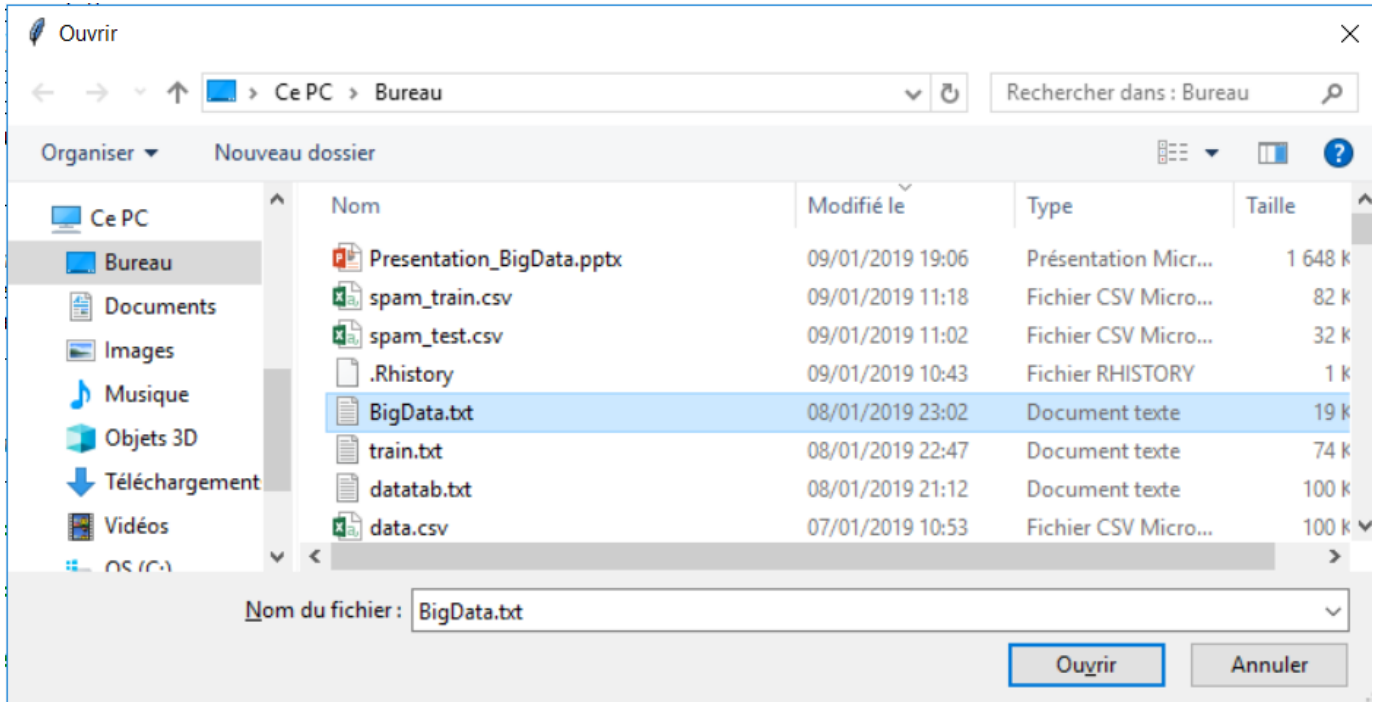
Matrice de confusion du Random Forest:



Matrice de confusion SVM :



Prédiction sur fichier à traiter

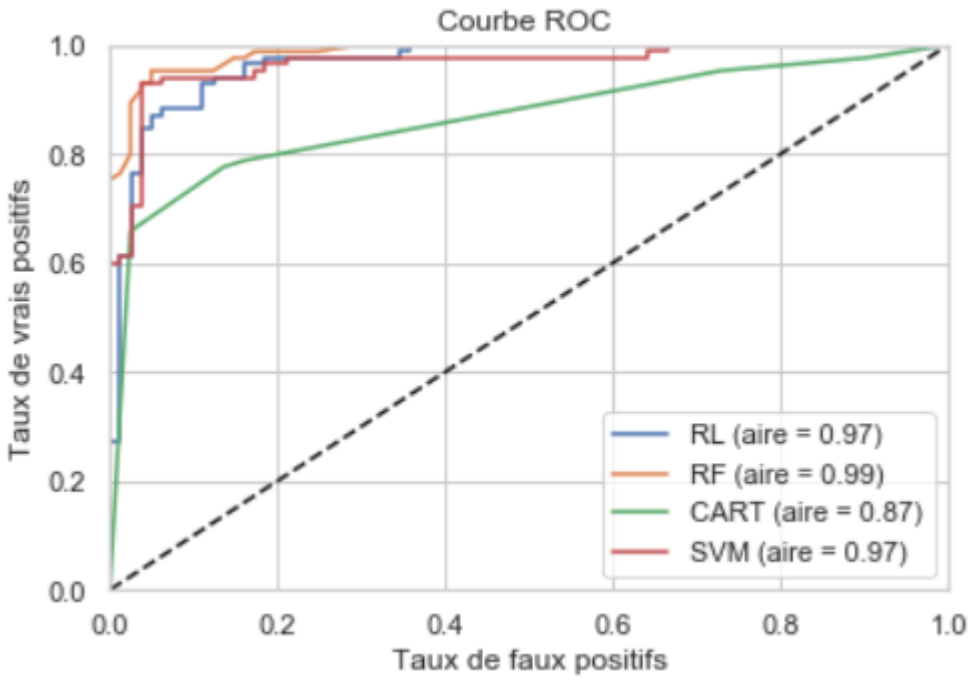


Enregistrer sortie au format csv

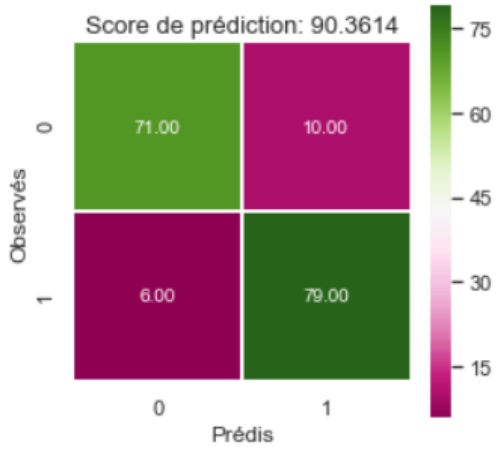
```
# On cherche la méthode sélectionné et on effectue la prédiction à partir de ce modèle
if ind == 0 :
    pred = logit_model.predict(dataset2)
elif ind == 1 :
    pred = rf_model.predict(dataset2)
elif ind == 2 :
    pred = cart_model.predict(dataset2)
else :
    pred = svm_model.predict(dataset2)
```

Reproductible ? Visualisation des résultats avec spam

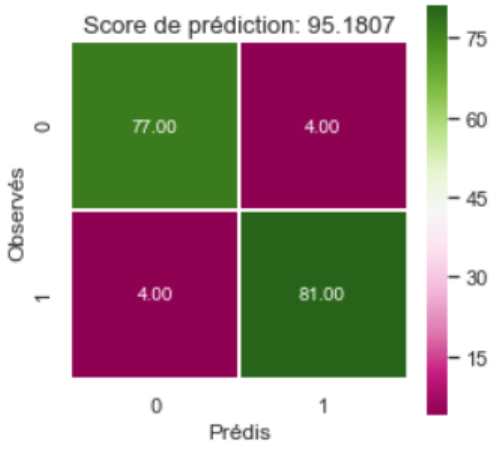
```
Précision de la Régression Logistique :  90.3614 %

Précision du Random Forest :  95.1807 %

Précision de l'arbre CART :  81.3253 %

Précision de SVM :  93.3735 %
```



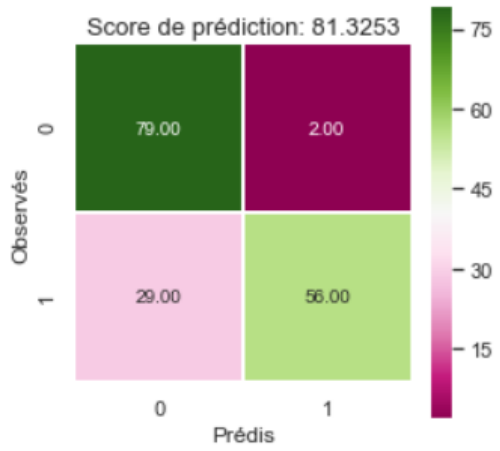
Matrice de confusion de la Régression Logistique:



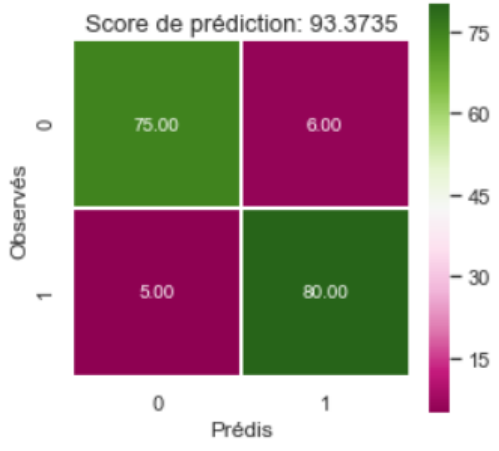
Matrice de confusion du Random Forest:



Matrice de confusion de CART:



Matrice de confusion SVM :

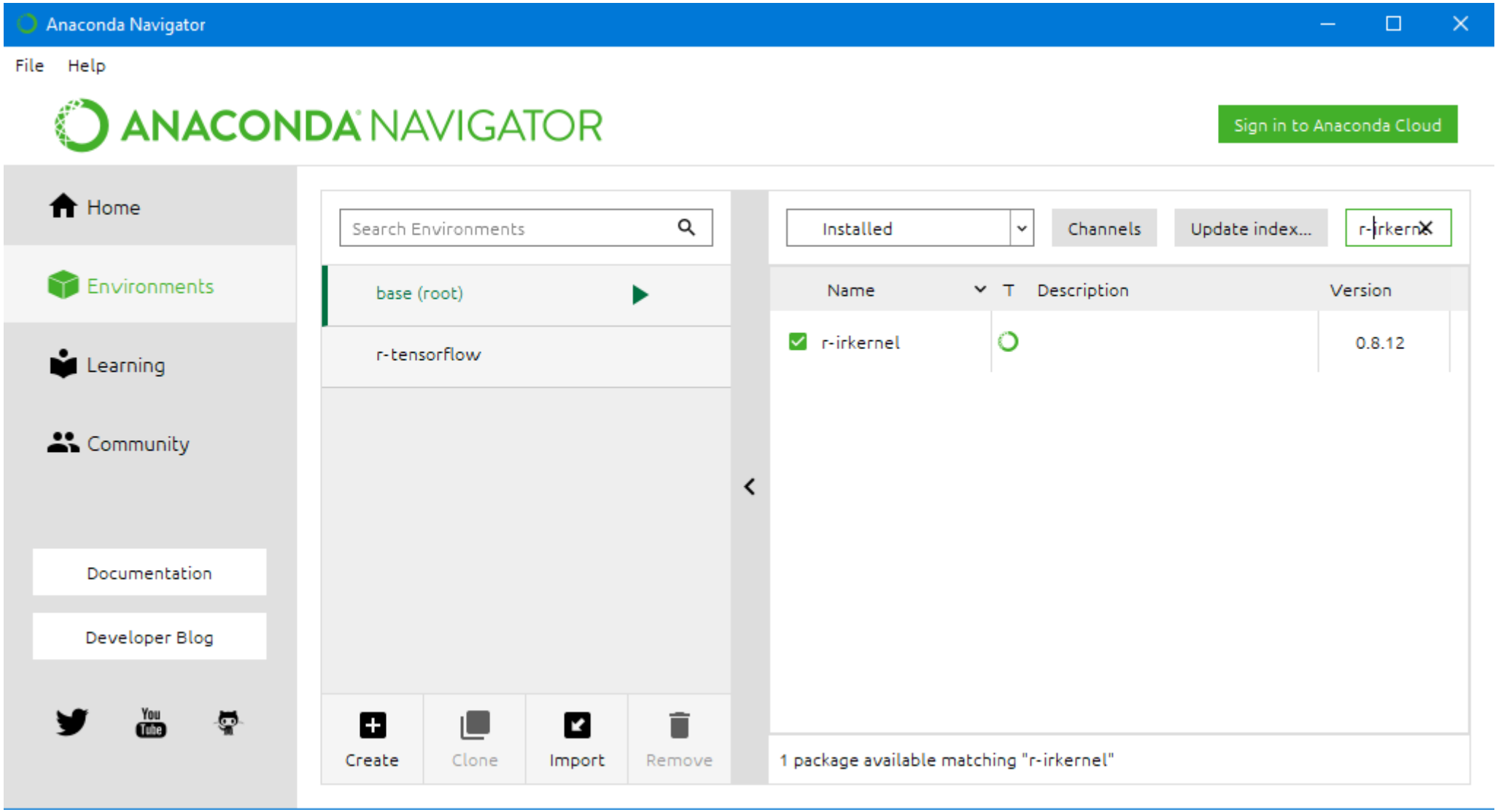


Jupyter avec R

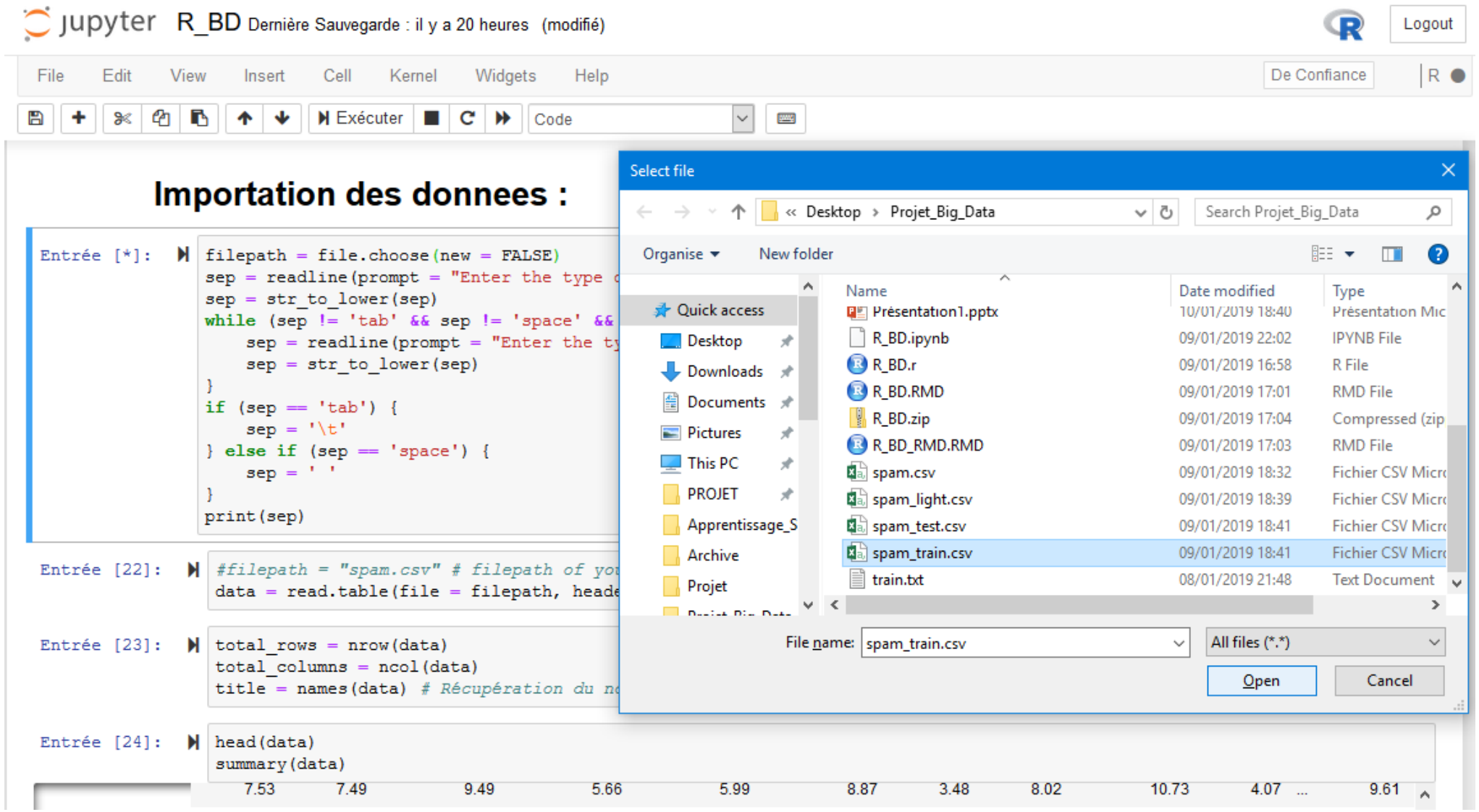
Installation IR Kernel

```
update.packages()
install.packages('IRkernel')
IRkernel::installspec(user = FALSE)
```

Installation IR Kernel



Script interactif : boîtes de dialogue



Script interactif : inputs

Importation des donnees :

Entrée [*]:

```
filepath = file.choose(new = FALSE)
sep = readline(prompt = "Enter the type of separator (;/,/tab/space) :")
sep = str_to_lower(sep)
while (sep != 'tab' && sep != 'space' && sep != ';' && sep != ',') {
    sep = readline(prompt = "Enter the type of separator (;/,/tab/space) :")
    sep = str_to_lower(sep)
}
if (sep == 'tab') {
    sep = '\t'
} else if (sep == 'space') {
    sep = ' '
}
print(sep)
```

```
Enter the type of separator (;/,/tab/space) : space
```

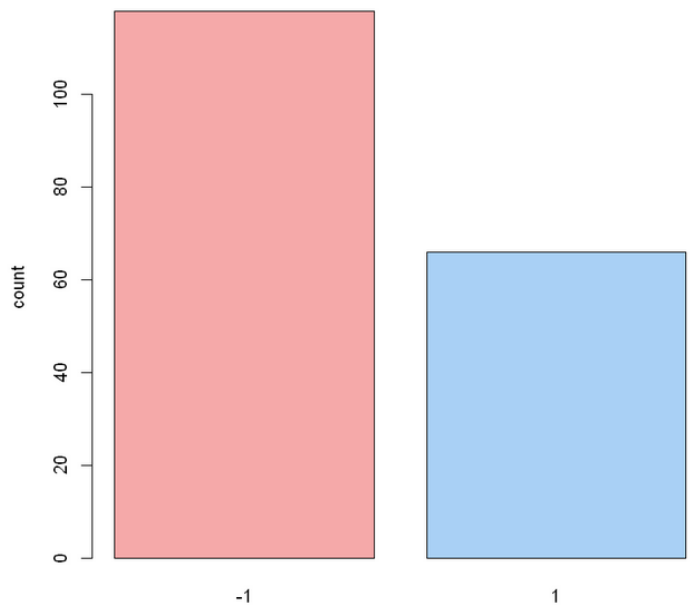
Analyse descriptive

Entrée [76]:
```
head(data)
```

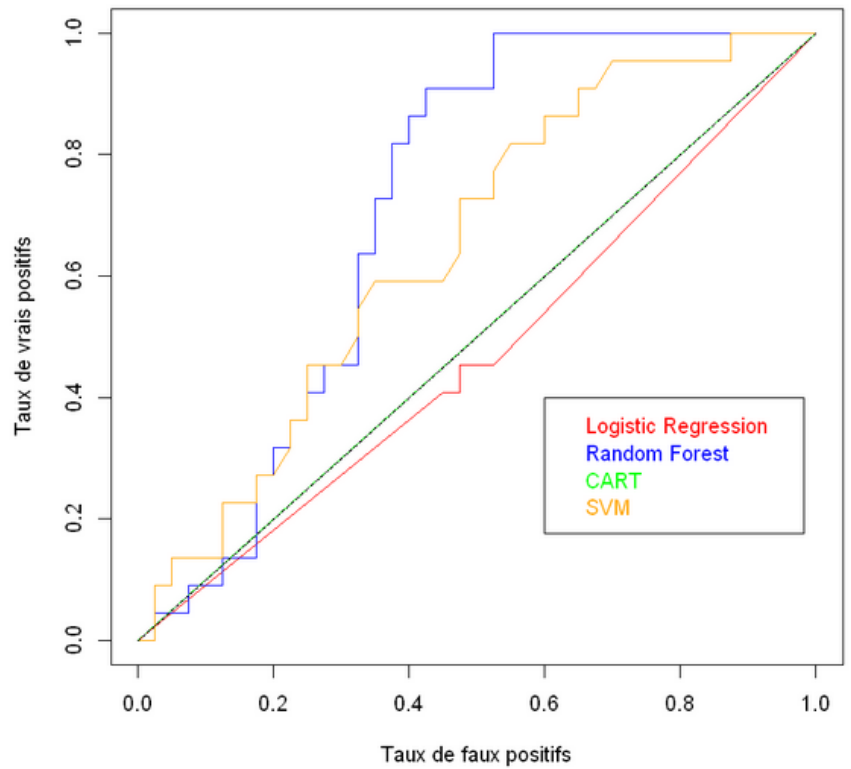
x10006_at	x10007_at	x100129361_at	x100130449_at	x100130633_at	x100133941_at	x10014_at	x10016_at	x100188893_at	x10019_at	...	x10226_at	x1
8.44	7.84	9.23	2.20	4.93	10.31	7.85	9.91	11.37	3.48	...	10.55	
7.93	8.47	7.77	2.48	6.70	11.03	3.33	9.53	10.91	4.28	...	9.00	
7.53	7.49	9.49	5.66	5.99	8.87	3.48	8.02	10.73	4.07	...	9.61	
7.72	8.38	8.03	5.22	6.95	2.73	4.06	9.26	10.90	7.40	...	9.91	
7.89	7.75	6.99	2.20	7.06	11.38	3.65	9.22	10.37	3.89	...	9.60	
8.37	8.43	8.32	2.20	7.02	10.70	3.17	7.73	11.18	4.41	...	8.76	

Répartition de la variable d'intérêt

Entrée [80]:
```
tbl <- with(data, table(y))
barplot(tbl, beside = TRUE, col = c("#F5A9A9", "#A9D0F5"), xlab = y, ylab = "count")
```



Courbes ROC

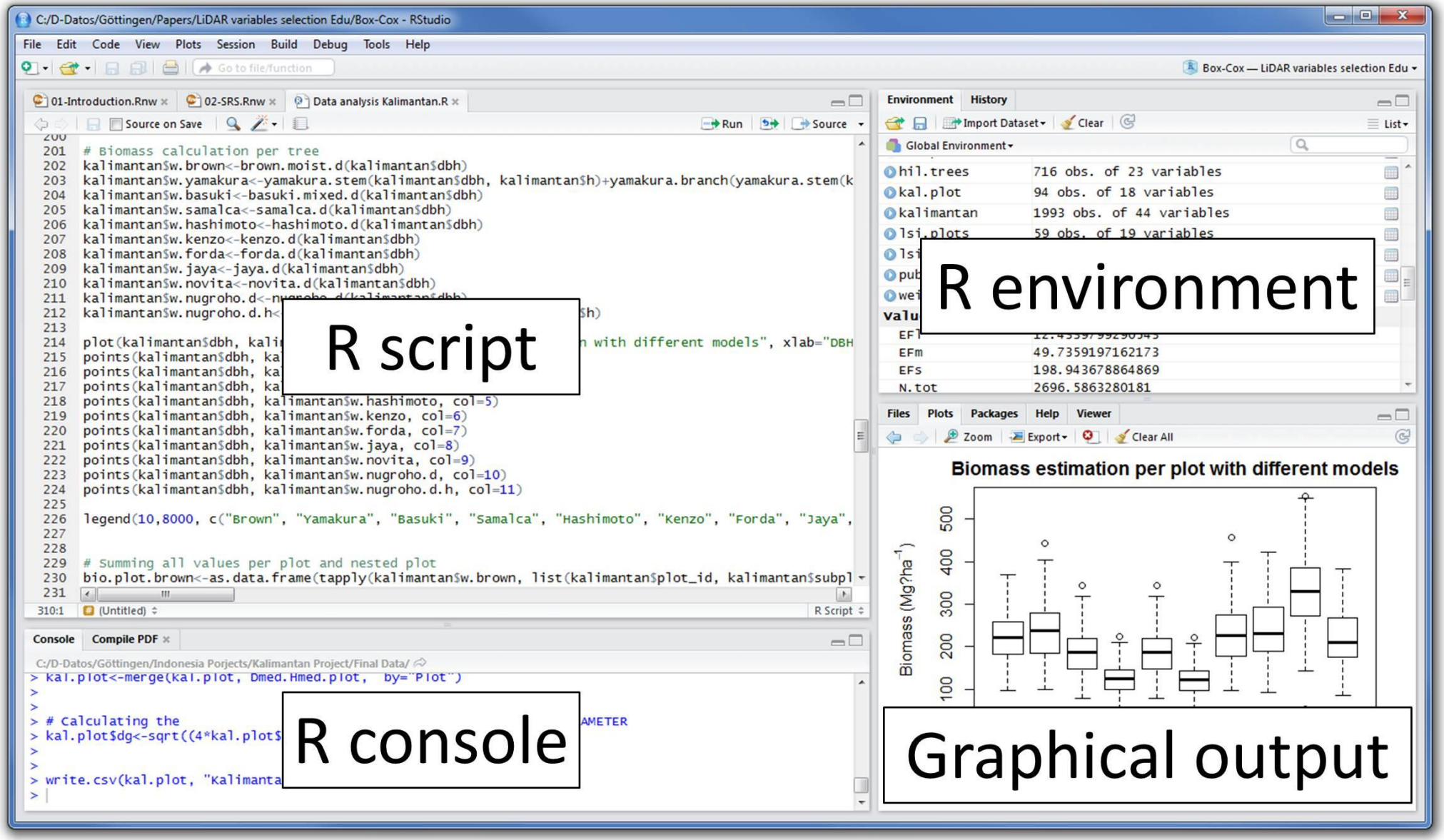


Avantages/inconvénients

- Portabilité
- Possibilité d’exporter le code R mais pas RMD
- Variables en mémoire non visibles

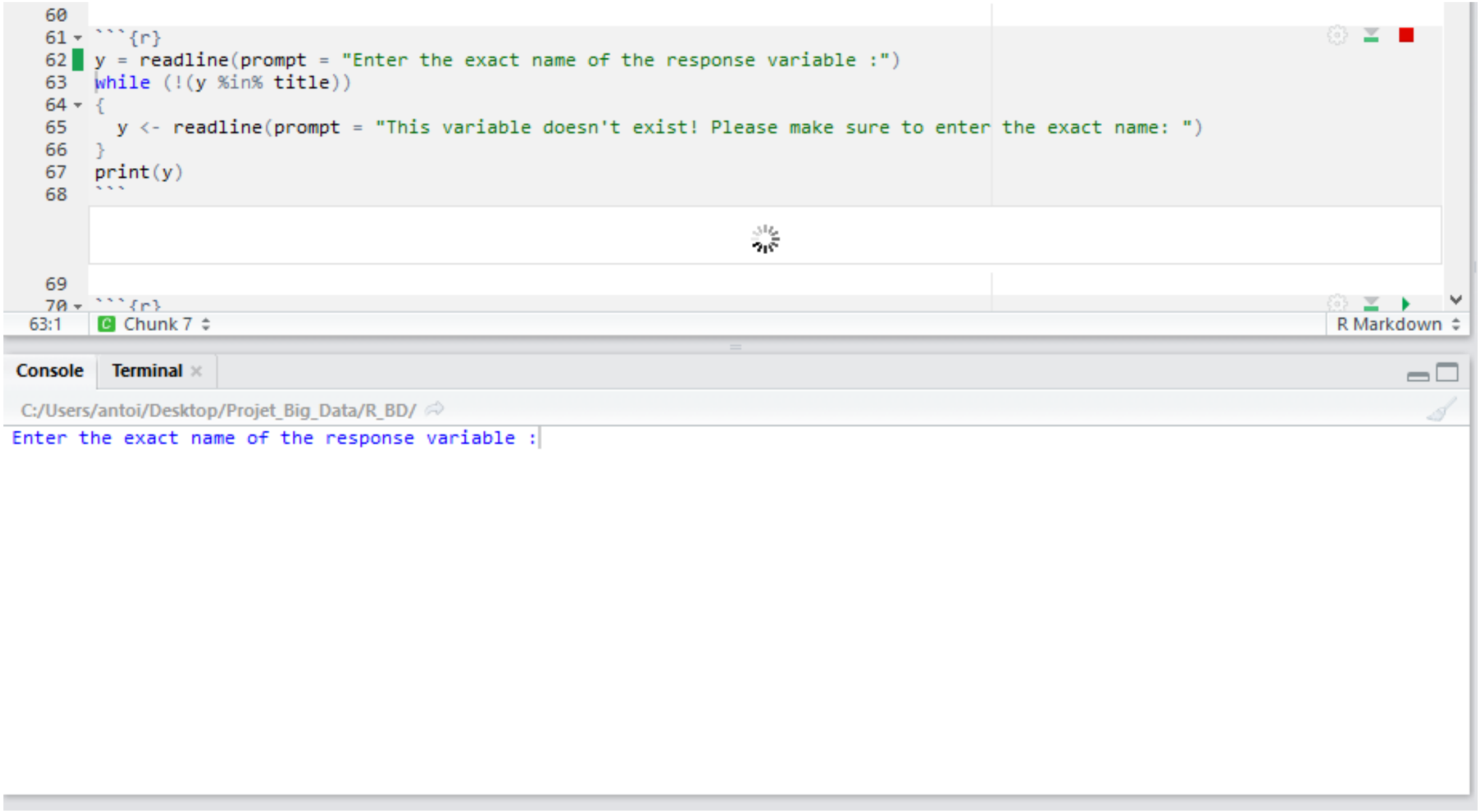
Présentation générale

Interface


C:/D-Datos/Göttingen/Papers/LiDAR variables selection Edu/Box-Cox - RStudio
File Edit Code View Plots Session Build Debug Tools Help
01-Introduction.Rnw
02-SRS.Rnw
Data analysis Kalimantan.R
Source on Save
Run
Source
Biomass calculation per tree
kalimantan$w.brown<-brown.moist.d(kalimantan$dbh)
kalimantan$w.yamakura<-yamakura.stem(kalimantan$dbh, kalimantan$h)+yamakura.branch(yamakura.stem(k
kalimantan$w.basuki<-basuki.mixed.d(kalimantan$dbh)
kalimantan$w.samalca<-samalca.d(kalimantan$dbh)
kalimantan$w.hashimoto<-hashimoto.d(kalimantan$dbh)
kalimantan$w.kenzo<-kenzo.d(kalimantan$dbh)
kalimantan$w.forda<-forda.d(kalimantan$dbh)
kalimantan$w.jaya<-jaya.d(kalimantan$dbh)
kalimantan$w.novita<-novita.d(kalimantan$dbh)
kalimantan$w.nugroho.d<-
kalimantan$w.nugroho.d.h<
plot(kalimantan$dbh, kali
points(kalimantan$dbh, ka
points(kalimantan$dbh, ka
points(kalimantan$dbh, ka
points(kalimantan$dbh, kalimantan$w.hashimoto, col=5)
points(kalimantan$dbh, kalimantan$w.kenzo, col=6)
points(kalimantan$dbh, kalimantan$w.forda, col=7)
points(kalimantan$dbh, kalimantan$w.jaya, col=8)
points(kalimantan$dbh, kalimantan$w.novita, col=9)
points(kalimantan$dbh, kalimantan$w.nugroho.d, col=10)
points(kalimantan$dbh, kalimantan$w.nugroho.d.h, col=11)
legend(10,8000, c("Brown", "Yamakura", "Basuki", "Samalca", "Hashimoto", "Kenzo", "Forda", "Jaya",
# Summing all values per plot and nested plot
bio.plot.brown<-as.data.frame(tapply(kalimantan$w.brown, list(kalimantan$plot_id, kalimantan$subpl
R script
Environment
History
Import Dataset
Clear
List
Global Environment
hil.trees 716 obs. of 23 variables
kal.plot 94 obs. of 18 variables
kalimantan 1993 obs. of 44 variables
lsi.plots 59 obs. of 19 variables
EFm 49.7359197162173
EFs 198.943678864869
N.tot 2696.5863280181
R environment
Files
Plots
Packages
Help
Viewer
Zoom
Export
Clear All
Biomass estimation per plot with different models
Biomass (Mg?ha-1)
100
200
300
400
500
Console
Compile PDF
C:/D-Datos/Göttingen/Indonesia Porjects/Kalimantan Project/Final Data/
> kal.plot<-merge(kal.plot, Dmed.Hmed.plot, by="Plot")
> # Calculating the
> kal.plot$dg<-sqrt((4*kal.plot$
> write.csv(kal.plot, "Kalimanta
R console
Graphical output

RStudio avec R

Avantages/Inconvénients



Rstudio avec Python

Inconvénient des inputs

```{python}
input2 = input("Indiquez la manière dont sont séparées vos variables (tabulation/espace/;) : ")
```

```
Indiquez la mani□re dont sont s□par□es vos variables (tabulation/espace/;) :
```

```{python}

# Importation du fichier
file = open("train.txt", "r")
dataset = pd.read_csv(file, sep=" ")

# Récupération du noms des différentes variables
title = list(dataset.columns)

# Variable d'intérêt
reponse = 'y'

```

Autre inconvénient

```{python}
x = 5
print(x)
```

```
5
```

```{python}
print(x)
```

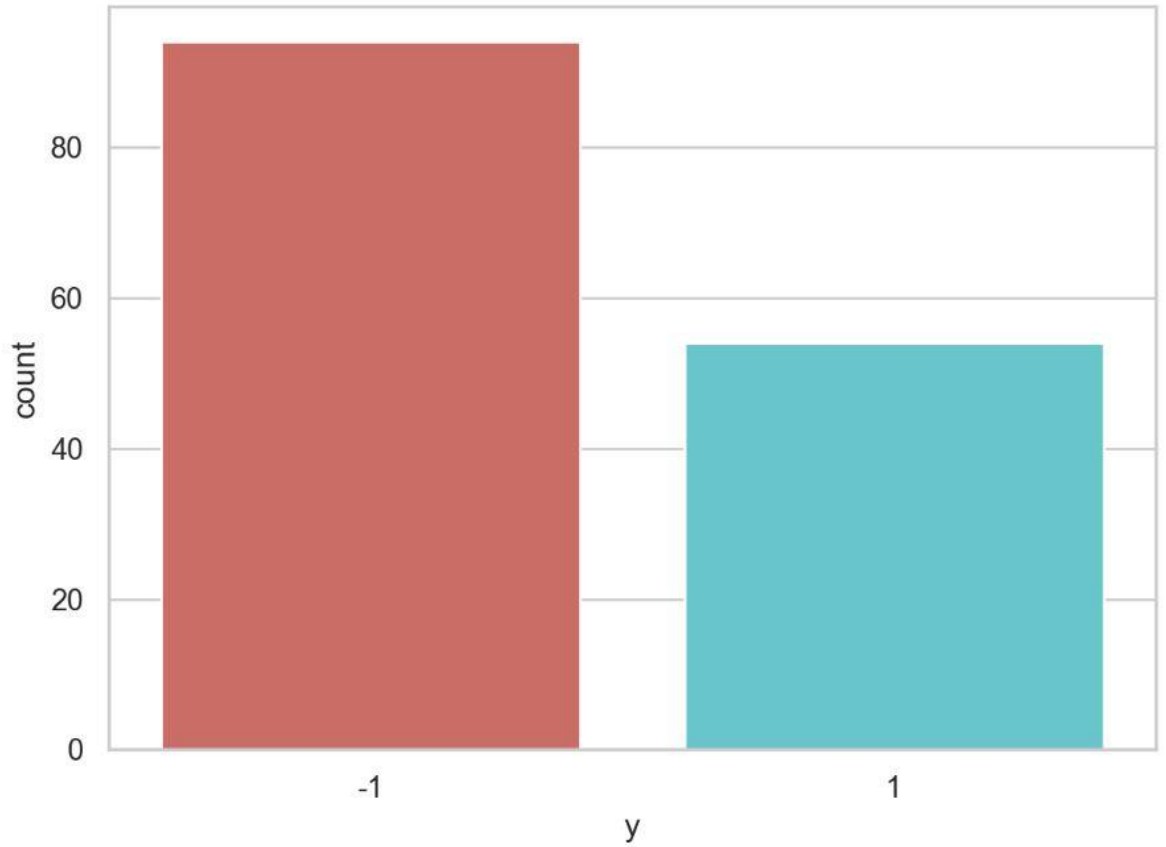
```
Traceback (most recent call last):
  File "C:\Users\marin\AppData\Local\Temp\RtmpI1mbbh\chunk-code-3bd07a3d62d.txt", line 1, in <module>
    print(x)
NameError: name 'x' is not defined
```

« Solution »

```
print("Nombre de classes : ", nb_classe)
```

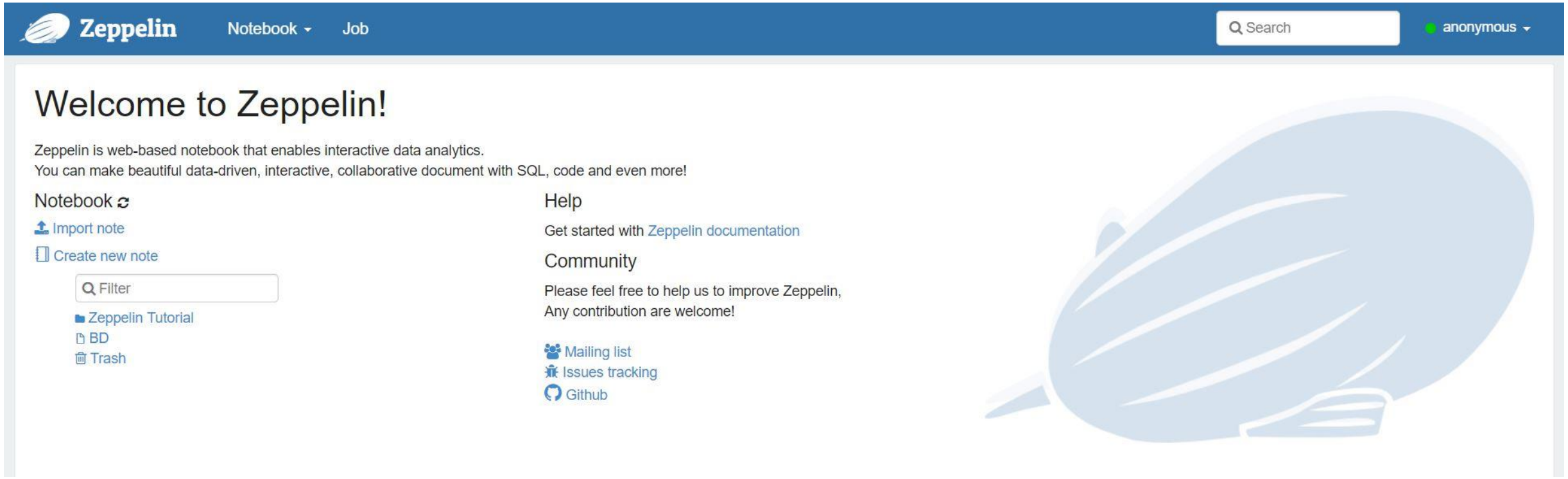
```
## Nombre de classes :  2
```

```
sns.countplot(x = reponse, data = dataset, palette = 'hls')
plt.show()
```



Présentation générale

Interface


Zeppelin
Notebook
Job
Search
anonymous
Welcome to Zeppelin!
Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!
Notebook
Import note
Create new note
Filter
Zeppelin Tutorial
BD
Trash
Help
Get started with Zeppelin documentation
Community
Please feel free to help us to improve Zeppelin,
Any contribution are welcome!
Mailing list
Issues tracking
Github

Exemple

Zeppelin Notebook Job Search anonymous

BD Head default

```
nb_classe =  len(list(set(y))) # Taille de la liste contenant la variable d'intérêt sans doublon
```

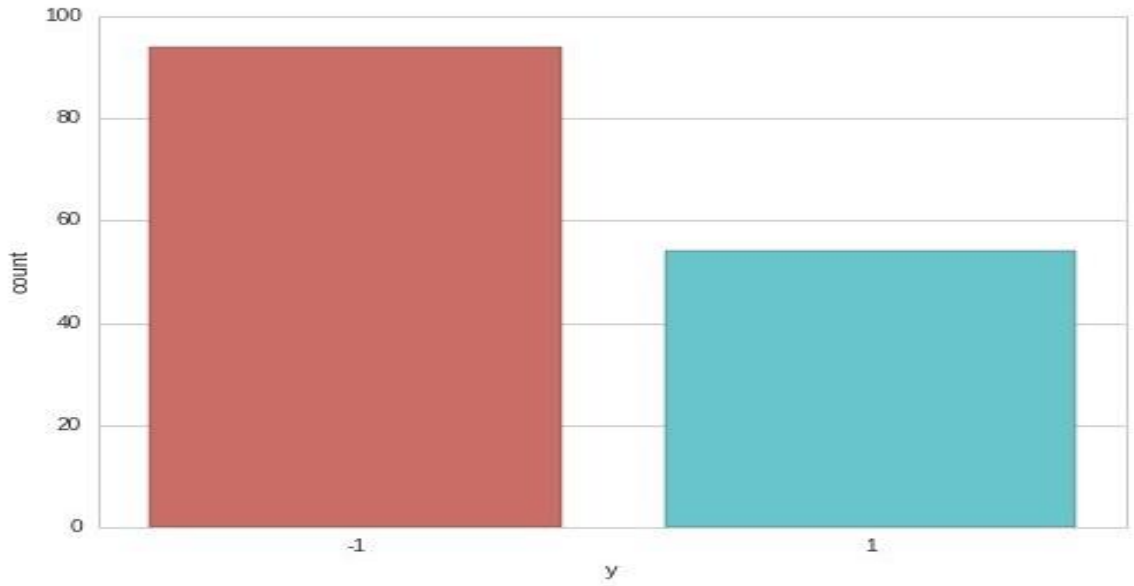
Interpréteur utilisé

Exécution de la cellule

```
%python
sns.countplot(x = reponse, data = dataset, palette = 'hls')
plt.show()
```

code

FINISHED



sorties

Took 0 sec. Last updated by anonymous at January 09 2019, 6:50:13 PM.

Zeppelin avec Python

Inconvénient des inputs

```
%python
input1 = z.input("Indiquez le chemin de votre fichier d'entrée : ")
#/home/tp-home008/mfigaro/Downloads/train.txt
```

Indiquez le chemin de votre fichier d'entrée :

/home/tp-home008/mfigaro/Downloads/train.txt

Took 0 sec. Last updated by anonymous at January 09 2019, 2:53:27 PM.

```
%python
input2 = z.input("Indiquez la manière dont sont séparées vos variables (tabulation/espace/;) : ")   # ici espace
# Passage en minuscule si l'utilisateur écrit en majuscule
input2 = input2.lower() ;
```

Indiquez la manière dont sont séparées vos variables (tabulation/espace/;) :

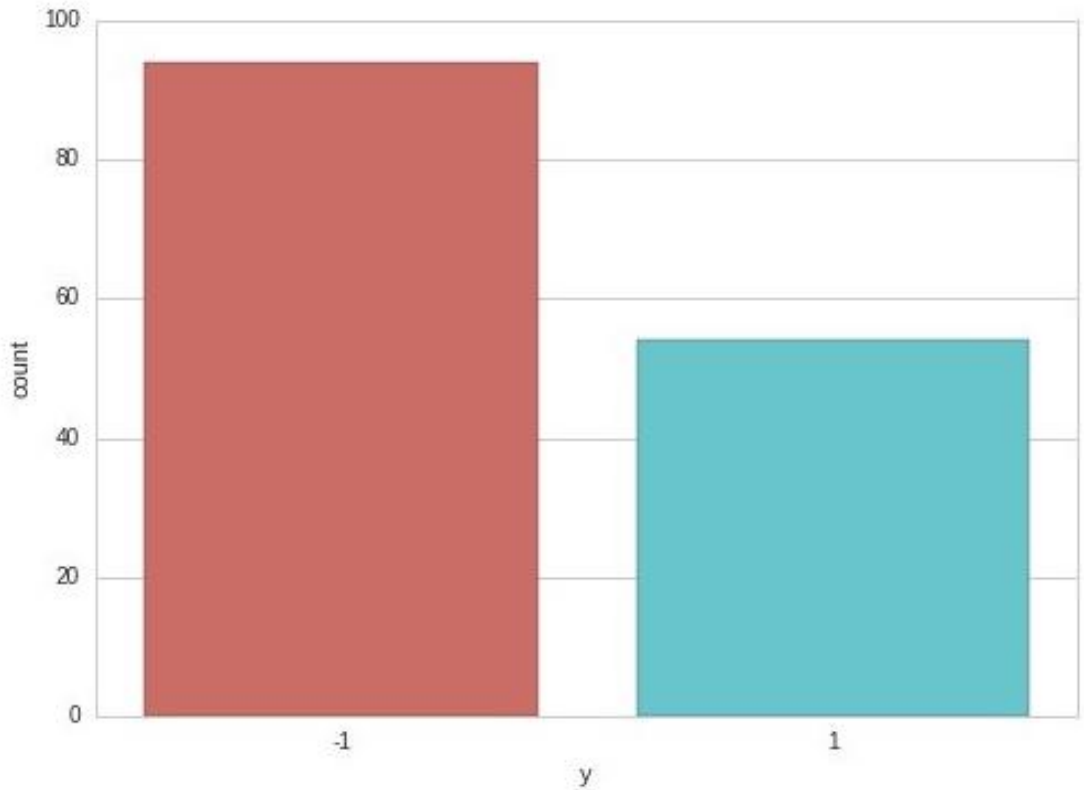
espace

Took 0 sec. Last updated by anonymous at January 09 2019, 2:53:27 PM.

```
%python
y.describe()
classe = list(set(y))
nb_classe =  len(list(set(y))) # Taille de la liste contenant la variable d'intérêt sans doublon

print "Differentes classes de la variable d'interet : ", classe
print "Nombre de classes : ", nb_classe
```

```
Differentes classes de la variable d'interet :  [1, -1]
Nombre de classes :  2
```

```
%python
sns.countplot(x = reponse, data = dataset, palette = 'hls')
plt.show()
```




```
%python
dataset.head()
```

	x10006_at	x10007_at	x100129361_at	x100130449_at	x100130633_at	\
0	8.44	7.84	9.23	2.28	4.93	
1	7.93	8.47	7.77	2.48	6.70	
2	7.53	7.49	9.49	5.66	5.99	
3	7.72	8.38	8.03	5.22	6.95	
4	7.89	7.75	6.99	2.20	7.06	

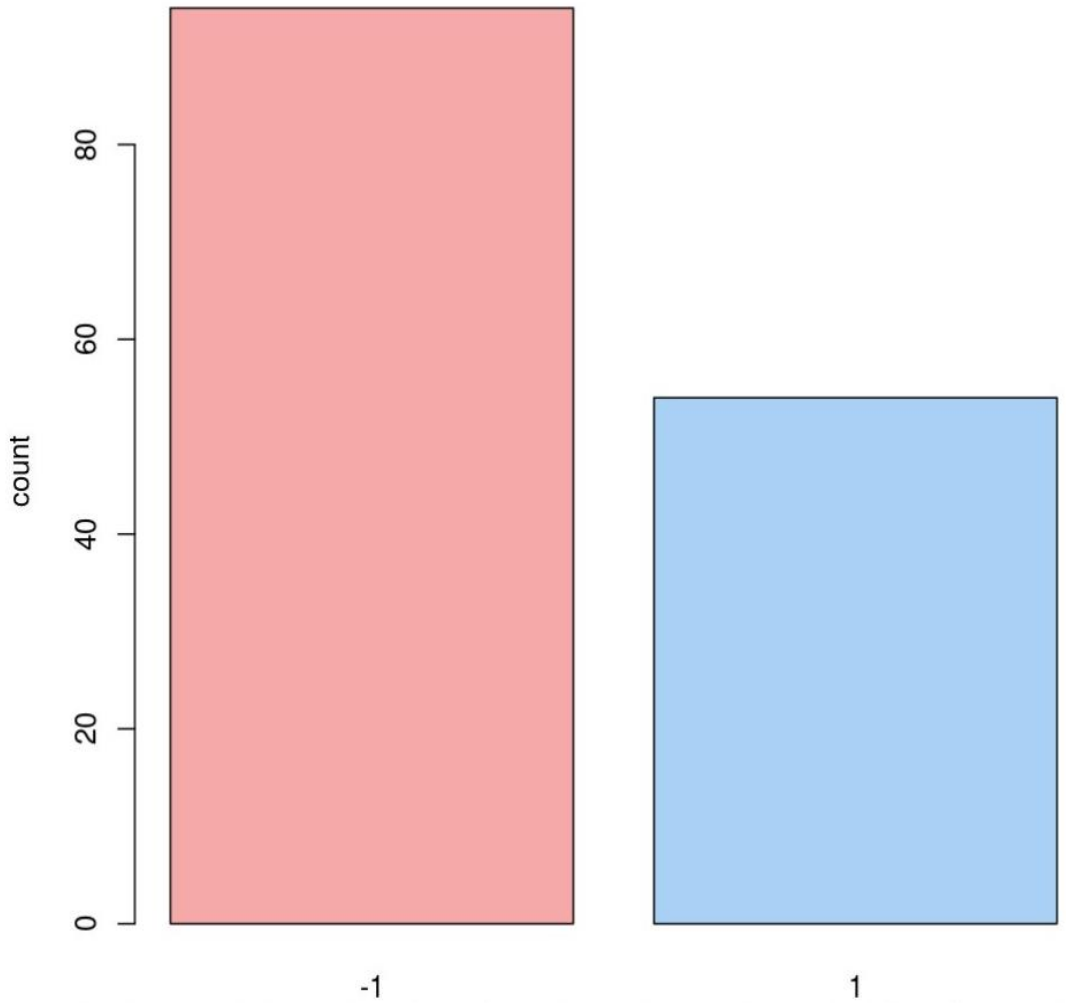
	x100133941_at	x10014_at	x10016_at	x100188893_at	x10019_at ...	\
0	10.31	7.85	9.91	11.37	3.48 ...	
1	11.03	3.33	9.53	10.91	4.28 ...	
2	8.87	3.48	8.02	10.73	4.07 ...	
3	2.73	4.06	9.26	10.90	7.40 ...	
4	11.38	3.65	9.22	10.37	3.89 ...	

	x10226_at	x10227_at	x1022_at	x10231_at	x10234_at	x10236_at	x10237_at	\
0	10.55	7.82	7.87	2.97	4.48	9.61	12.82	
1	9.00	7.88	7.38	3.31	6.69	9.50	7.99	

Zeppelin avec R

```
%spark.r
tbl <- with(data, table(y))
barplot(tbl, beside = TRUE, col = c("#F5A9A9", "#A9D0F5"), xlab = y, ylab = "count")
```

FINISHED




```
%spark.r
head(data)
summary(data)
```

```
x10006_at x10007_at x100129361_at x100130449_at x100130633_at
1      8.44      7.84          9.23          2.20          4.93
2      7.93      8.47          7.77          2.48          6.70
3      7.53      7.49          9.49          5.66          5.99
4      7.72      8.38          8.03          5.22          6.95
5      7.89      7.75          6.99          2.20          7.06
6      8.37      8.43          8.32          2.20          7.02
  x100133941_at x10014_at x10016_at x100188893_at x10019_at x1001_at
1         10.31      7.85      9.91         11.37      3.48     4.89
2         11.03      3.33      9.53         10.91      4.28     5.81
3          8.87      3.48      8.02         10.73      4.07     5.21
4          2.73      4.06      9.26         10.90      7.40     3.27
5         11.38      3.65      9.22         10.37      3.89     7.18
6         10.70      3.17      7.73         11.18      4.41     8.40
  x10020_at x10026_at x100272147_at x100287025_at x100287552_at
1      5.61      5.61          8.64          6.15          5.71
```

Conclusion

	Jupyter	RStudio	Zeppelin
Type d'application	Web	Web, Logiciel	Web
Nombre langages supportés	+++	+	++
Installation	Facile	Facile	Difficile
Prise en main	Facile	Facile	Facile
Documentation	+++	+	-
Support	+++	++	+
Format d'export	++	+	-