R examen

Lucas Billaud

janvier 2021

Critères d'évaluation :

- Explication du code
- Exemples utilisés
- Utilisation du package
- Finalité de l'utilisation
- La qualité de rédaction

Travaux 1

GGPLOT2
par Maxime et Siva
https://github.com/Siva-chane/PSBX/blob/main/package%20ggplot2/Package_ggplot2.pdf

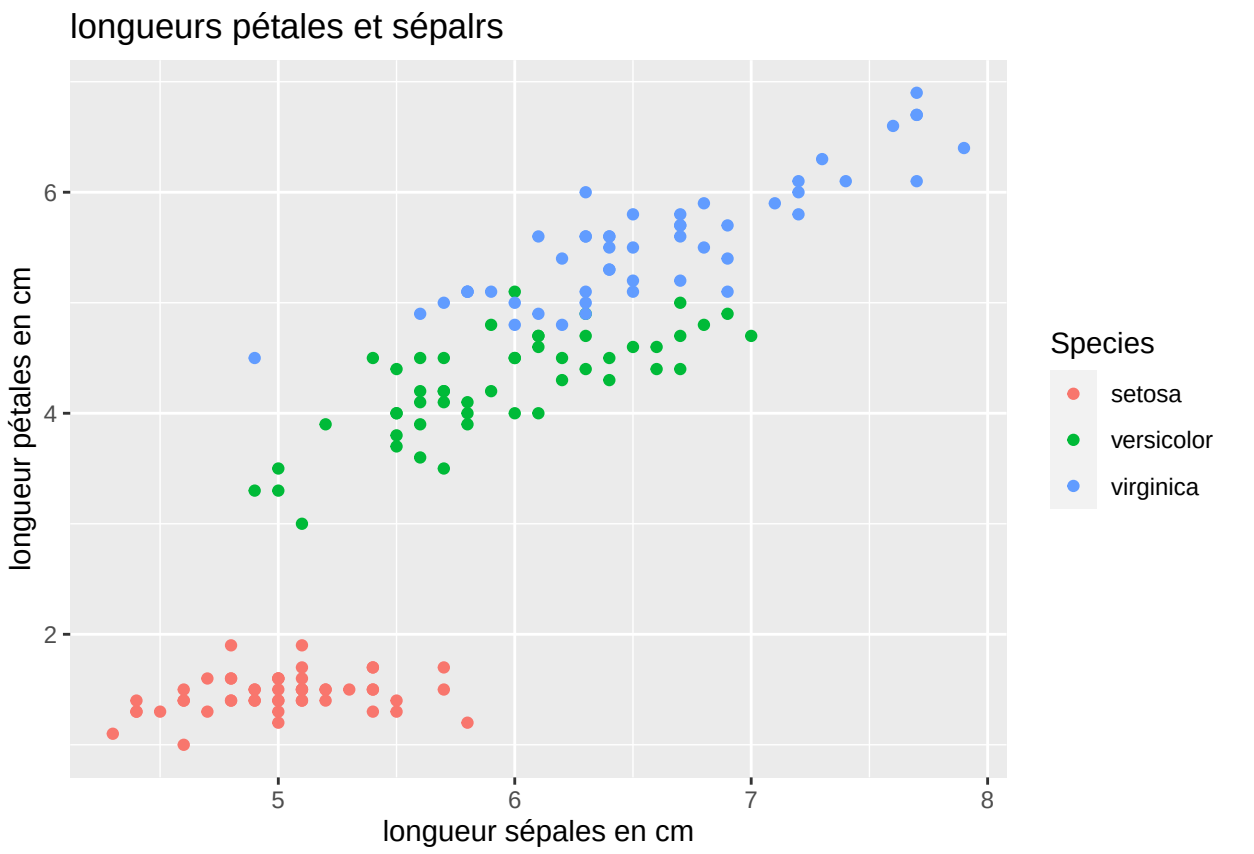
A.Synthèse

A travers GGplot2 on aborde l'une des parties de l'analyse de données : "l'exploration des données". GGPLOT2 s'y prête bien car étant un package de visualisation. Il permet de représenter graphiquement les données avec des visuels parlant. En effet, on peut déjà tirer beaucoup d'informations de ces données à l'aide des visualisations, ainsi ggplot 2 étant basé sur le principe de "grammaire des graphiques" permet de faire ressortir les traits de caractère des données.

B. Création et explication de création de certains graphiques.

```
library(ggplot2)
data(iris)
graph_ <- ggplot(iris, aes(Sepal.Length, Petal.Length,
                           colour=Species))+geom_point()

print(graph_ + labs(y="longueur pétales en cm", x="longueur sépales en cm")
      + ggtitle("longueurs pétales et sépalrs"))
```



Afin de créer des graphiques avec ggplot on utilise un mot clé et on passe ensuite les informations que l'on souhaite visualiser. Dans la visualisation d'en haut le premier arguement correspond au dataset utilisé. Ensuite on spécifie les variables que l'on souhaite visualiser. La première sera en abscisse et la seconde en ordonnées. On peut aussi spécifier les différenst groupes. Dans notre cas, nous utilisatons colour=Species afin d'affecter à chaque espèce une couleur différente. Enfin geom_point() permet de doser la taille des points. Il est ensuite possible de modifier les légendes et titres dans un print.

C. Evaluation

Le cadre de l'utilisation de ggplot2 pour l'exploration des données est bien posé. Tout au long du document on a une approche assez progressive en commençant par des graphiques conventionnels en allant vers des représentations de plus en plus visuelles. L'explication de l'utilisation du package est claire et les exemples sont concrets. De plus chaque type de graphique est commenté et là on met en avant le code utilisé.

D. Conclusion

A travers ce tutoriel d'utilisation, on voit les différentes façons qui s'offrent à nous afin de visualiser les données. Il est aussi montré comment affiner de plus en plus un graphique en ajoutant des éléments visuels.

Travaux 2

R vs Python
par Ren Claude
https://github.com/Cldren/REN_PSBx/blob/main/R_Docs/R-vs-Python.pdf

A. Synthèse

Dans ce document “R vs Python” nous abordons les différences basiques entre les 2 languages. Dans un 1er temps, nous voyons les nuances sur les opérateurs par exemple, python utilise des packeges pour certaines opérations mathématiques alors que dans R, ceux-ci sont déjà intégrés. Les 2 languages permettent de faire beaucoup de choses et ont leurs propre caractéristiques (synthaxique) qui sont mises en avant à l'aide de ce papier comparatif.

B. Focus sur les différences.

R

```
pi
```

```
[1] 3.141593
```

```
w <- c(1, 2, 3)
v <- c(10, 20, 30)
w-v
```

```
[1]  -9 -18 -27
```

python

```
import numpy as np

np.pi
```

```
3.141592653589793
```

```
v = [1,2,3]
w = [10,20,30]
try :
  v-w
except :
  print("error")
```

```
error
```

```
w = np.array([1,2,3])
v = np.array([10,20,30])
```

Afin de traiter certaines choses python doit utiliser des packages alors que R non. C'est le cas pour la lecture de certains nombre mathématiques et pour les opérations entre les listes (il en existe d'autres mais en faire la liste serait longue). Concernant les listes python n'intègre pas naturellement les opérations il faut appeler numpy afin de pouvoir le faire sans utiliser la boucle. D'un point de vue synthaxique on a tendace à utiliser “<-” pour R ce qui correspond à “=” en python et qui représente une affectation de variables.

C. Evaluation du travail

Les explications sur la différence des 2 languages sont très claires. De plus on voit directement les caractéristiques de chacun d'entre eux. Nous avons des exemples très concrets sur la synthaxe et les nuances des 2 languages. Cela permet d'avoir une première vue d'ensemble sur les 2 languages. Enfin la présentation des 2 codes étant faite à l'aide de colonnes correspondantes à R et python en face à face facilite la compréhension.

D. Conclusion

Ce travail nous apporte une vue Globale de l'utilisation synthaxique de python et R. En soulevant l'utilisation des packages dans certains cas pour python alors que R intègre nativement les propriétés mathématiques.

Travaux 3

DLYPR
par Fontaine Grégoire
https://github.com/gfontainepsb/Cours-R/blob/main/dplyr.pdf

A.Synthèse

Dans ce tutoriel, nous traitons le package dplyr. En partant du principe que les données soient déjà “tidy”, dplyr est utilsé afin de manipuler les dataframes. Sa synthaxe s’appuie sur des verbes afin de pouvoir par exemple selectionner, filtrer... Ce tutoriel est une introduction à ce package et montre les fonctions de base de dplyr.

B. Explication des fonctions

```
library(dplyr)
```

```
Attaching package: 'dplyr'
```

```
The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
head(iris) %>%
 filter(Species == "setosa")
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

```
head(iris) %>%
 mutate(Species, ratio_sepal_lenght_widdth = Sepal.Length/Sepal.Width)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
  ratio_sepal_lenght_widdth
1                  1.457143
2                  1.633333
3                  1.468750
4                  1.483871
5                  1.388889
```

```
6                  1.384615
```

On présente ici les fonctions filter qui permettent de filter par rapport aux valeurs. Par exemple nous ne prenons en compte que les setosas. L'option mutate quant à elle permet de d'ajouter des colonnes/de créer des colonnes suite à des opérations sur d'autre colonnes. On peut aussi combiner plusieurs actions pour avoir des "filtres" plus complexes.

C Evaluation

Le tutoriel est facile à comprendre. On aborde les fonctions de base de dplyr sans entrer dans un schéma trop complexe. On a 3 exemples sur une dataframe. Et les fonctions abordées s'expliquent d'elles mêmes car elles ont une syntaxe commune au langage naturel.

D. Conclusion

Ce tarvail aborde quelques fonctions basiques de dplyr. Ce package est utilisé afin de manipuler des dataframes.

Travaux 4

Keras
Par Kabirou Kanlanfeyi, Jordy Hounsinou
https://github.com/kabirou7/PSBX/blob/main/Keras_FashionMNIST.pdf

A. Synthèse

Dans ce tutoriel nous abordons la bibliothèque Keras. Keras permet d'intéragir avec des alogrithmes de réseaux de neurones profonds. Bien qu'elle soit initiallement écrite en python, elle peut être utilisée dans R. Ce tutoriel nous montre l'implémention d'un reseau neurones sur la base de données MNIST Fashion. Dans cet exemple Keras a été utilisé afin de reconnaître des images de vêtements.

B. Les grandes lignes de l'implémenation.

Ayant rencontré un problème d'installation de keras, nous mettrons les lignes principales du code en commentaire.

En considérant que les données soient importées et séparées en sets d'entraînement et en sets de test on a :

Une partie prétraitement et normalisation des données. Travaillant avec des images, on divise chaque colonne par 255 (nombres de pixels max)

```
#train_images <- train_image / 255
#train_test <- train_test / 255
```

On crée ensuite le réseau neuronal en partant d'un réseau vide. Et on ajoute au fur et à mesure les différentes couches afin de créer le réseau.

```
#model <- keras_model_sequential
#model %>%
  #layer_flatten
  #layer_dense
  #layer_dropout

#la partie des layers peuvent être répéter
# cela dépend du nombre de couche souhaité.
```

On passe ensuite à la partie compilation et entrainement du reseau.

```
#model %>% fit
```

puis la partie prédiction

```
#prediction <- model %>% predit_classes()
```

C. Evaluation

Ce sujet utilise un exeple concret pour l'implémentation du réseau neuronal avec Keras. Le code est globalement bien expliqué, même si certains points restent flous (je n'ai pas compris la partie pourquoi il faut compiler le modèle et la partie matrice de confusion). Cependant, on comprend comment on peut utiliser keras (ici classification/reconnaissance des images). Et j'ai pu découvrir des notions que je ne maitrise pas encore comme fonction d'activation, le fonctionnement des couches dans le résau (à creuser).

Conclusion

En conclusion Keras est une bibliothèque dédiée au deep learning. Elle permet notamment de faire de la reconnaissance d'images en transformant les images en vecteur de nombres. Son implémentation a l'air simple d'utilisation lorsque l'on maîtrise le sujet.

Travaux 5

Data.table
Par Claire MAZZUCATO et Adrien JUPITER
https://github.com/clairemazzucato/PSBX/tree/main/Packages/Data.table

A. Synthèse

Dans ce tutoriel on aborde le package data.table. Il permet aussi comme dplyr de manipuler les dataframes appeleés ici data table (j'ai cependant l'impression qu'il y a plus de fonctions dans data.table en comparant avec le tuto sur dplyr). Data.table est cependant plus radipe/performant que dplyr et permet de traiter des tableaux plus grands. La synathaxe a l'air très proche de pandas (python).

B. Exploration du package

```
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
#transforme en datatable
#mtcars_dt = setDT(mtcars)

#head(mtcars_dt[, 1, with=F])

#mtcars_dt[, .(mean_mileage=mean(mpg)), by=cyl]
```

Afin de pouvoir utiliser les fonctions de data.table il faut d'abord convertir les dataframes en datatable. Il y a deux méthodes l'une qui crée une copie, l'autre qui transforme la dataframe déjà existante. De plus on voit certaines nuances que possède le package. Par exemple on ne peut pas sélectionner les colonnes avec le numéro de leur position sans préciser with=F. Aussi la façon de grouper par rapport aux valeurs est un peu plus intuitive que la fonction "aggregate()" de base. Le plus souvent data.table utilise la même forme afin de réaliser les actions :
datatable[row, columns, group]

Evaluation

Le tutoriel est clair, on aborde plusieurs fonctions de datatable en donnant son réel avantage sur les autres packages de manipulation. On montre aussi bien les nuances que le package possède notamment dans le cadre de lecture d'un csv. Les exemples utilissé sont concrets; de plus le schéma récapitulatif de fonctionnement de data.table [row,column,group] permet de comprendre l'intuition de son utilisation.

Conclusion

Le package data.table est utile pour les grandes tables de données. En effet, sa rapidité d'exécution est plus performante. De plus, elle possède une synthaxe simple afin de mieux s'approprier son utilisation.

Auto évaluation package Porphet

Prophet https://github.com/Lucasblld/PSBX/tree/main/R%20packages

J'ai présenté le package Prophet qui est utilisé pour la prédiction de Série temporelle. Ce tutoriel est simple d'accès et montre l'implétation d'une prédiction simple. J'ai aussi éssayé d'aborder de façon assez générale les mathématiques pour expliquer le fonctionnement du package. IL aurait été souhaitable de partir un peu plus loin en montrant les différents paramêtre qui peuvent être utilisés dans prophet. Il aurait été aussi bien d'aborder les différentes colonnes qui sont créees lorsque l'on fait la prédiction. Enfin ayant fait le tutoriel sur Rcloud, je n'ai pas eu de problème d'installation. Cepedant je me suis rendu compte après coup que certaines manipulations étaient nécéssaires afin de l'installer en local (chose que je n'ai pas traité).