République du Sénégal

Un peuple - Un but - Une foi

Agence Nationale de la Statistique et de la Démographie (ANSD)

Ecole Nationale de la Statistique et de l'Analyse Économique (ENSAE)

PROJET STATISTIQUE SUR R ET PYTHON

RÉSUMÉ DES EXPOSÉS DES GROUPES DE R

Rédigé par :

LOUGUE Nicolas

SOSSOU Toussaint Régis

AMEWOUAME Mawuéna Elisée

NGATCHA NOUTCHA Stephy Nadia

Élèves Ingénieur Statisticien Économiste

Sous la supervision de :

M. HEMA Aboubacar

Analyst researcher

Juillet 2023

Introduction

Le présent document fait la synthèse des différents exposés suivis dans le cadre du cours de *Projet Statistique sous R et Python* dispensé par **M. Hema Aboubacar** à L'ENSAE de Dakar.

Installation d'un package

Pour utiliser un package sur R, il est nécessaire de l'installer et ceci se fait avec la commande :

```
install.packages("nom_du_package")
```

- Exemple du package *janitor*

```
install.packages("janitor")
```

1 Janitor

Janitor « homme à tout faire »est un package de R qui permet de faire toute sorte d'opérations sur les bases de données. Il est particulièrement utile pour le traitement des bases de données issues d'enquêtes notamment dans le nettoyage et l'exploration de celles-ci. Entre autre, il permet de traiter les valeurs manquantes, de créer et de recoder les variables, de supprimer les doublons, de créer des variables dérivées et de changer le format des noms des variables.

1.1 Quelques fonctions de Janitor

Pour effectuer toutes ces opérations Janitor utilise plusieurs fonctions dont les plus importantes sont :

- *clean_ names()* : elle sert à renommer des variables de manière automatique.
- *remove_ empty()* : Elle permet de supprimer une ou plusieurs colonnes qui ne contiennent que des valeurs manquantes.
- *get_ dupes()* : Elle permet de détecter les doublons dans une base de données.
- *remove_ constant()* : Elle permet de supprimer les variables constantes (qui ne contiennent qu'une seule valeur) de la base de données.
- *make_ clean_ names()* : Elle sert à renommer les variables en fonction des préférences de l'utilisateur.
- *compare_ df_ cols()* : Elle permet de résumer les différentes comparaisons entre les colonnes de deux dataframes.

1.2 Limites de janitor

Le package janitor dépend d'autres packages pour certaines de ses fonctionnalités. Cela signifie que l'on doit s'assurer d'avoir ces packages supplémentaires installés et chargés pour pouvoir utiliser pleinement les fonctionnalités de Janitor. Janitor ne peut pas faire le croisement de 3 variables contrairement à gtsumary.

2 gtsumary

Le package gtsummary a été développé à des fins d'évaluation d'impact et permet de faire des tableaux statistiques et des tris à plat. Il génère des tableaux (tableaux de contingence, croisement de plusieurs variables, résumé des variables) résumant les données et permettant de faire des comparaisons entre les groupes (« cas témoin » versus « traitement » dans le cas de deux groupes), avec la possibilité de faire des résumés de plus de deux groupes de variables. Il intègre la possibilité d'exporter directement ses tableaux sous *Word*, en format *pdf* et sous *html* notamment avec les packages *« flextable »* et *« gt »* pour l'exportation des résultats.

2.1 Quelques fonctions de gtsummary

- *tbl_summary()* : Crée une table de **résumé** pour la variable spécifiée, qu'elle soit binaire, continue, multichotomique.
  Quelques paramètres de la fonction *tbl_summary()*
  - *label=list(nom_de_variable ~ "Nouveau_nom")* : pour renommer le nom des variables dans le tableau de sortie ; cela est particulièrement utile pour les noms de variables inadéquats ou trop long.
  - *by=nom_de_variable* : affectation de la variable servant à la formation des groupes
  - *missing_text = "Missing"*, pour l'affichage des valeurs manquantes dans le tableau de sortie.
- *add_difference()* : pour faire une évaluation d'impact, en comparant les différences entre le cas témoin et le traitement.
- *tbl_cross()* : Pour faire les tableaux de contingence.
- *flextable : :save_as_docx(variable_resultat,path = "chemin_voulu")* : il s'agit d'une fonction du package *flextable* qui permet d'exporter le contenu de la *variable_resultat* issue de l'affectation du contenu d'un tableau *gtsummary* dans un fichier word dont le chemin d'accès est indexé par *« path »*.

2.2 Dépendance avec d'autres packages

Comme nous pouvons déjà le constater, le package *gtsummary* est un package qui est assez dépendant d'autres packages comme *dplyr* pour la sélections des colonnes de données depuis la base de donnée, du package *flextable* pour l'exportation des tableaux sous différents formats.

3 R Markdown

R Markdown est une extension de R qui permet de créer des documents dynamiques avec R sous format **html**, **pdf**, **word**, **powerpoint**, **beamer**. Il sert souvent à créer des rapports dynamiques avec des codes R intégrés ou non (selon les préférences de l'utilisateur). Cela facilite le travail du statisticien en lui permettant de générer des rapports actualisés avec de nouvelles données.

3.1 Création d'un fichier R Markdown

Pour la création d'un fichier R Markdown on peut procéder comme suit :

- *File* ensuite,
- *New file* puis,
- *R Markdown* puis s'ouvre une fenêtre où on peut mettre les métadonnées du document et spécifier le format de sortie et on appuie *ok* et le fichier sera créé.

Une fois que la rédaction du document est terminée, on pourra cliquer sur le bouton *knit* ou faire *Ctrl+Shift+K* pour générer le document.

3.2 Les différentes parties d'un document *R Markdown*

Un document R Markdown contient essentiellement 3 parties :

- Une entête YAML : qui contient les métadonnées du document c'est-à-dire l'auteur, le titre, la date de création du fichier. Dans cette partie on peut aussi ajouter des paramètres de mise en forme du document comme le thème (format de sortie) de celui-ci, l'extension du fichier de sortie.
- La partie texte : c'est la partie du document où on met effectivement le texte à écrire. Il y a différents codes qui permettent de le mettre en forme (gras, italique, souligné, surligné...) afin de produire un joli document.
- Les chunks (blocs de codes) : ils peuvent être placés n'importe où dans la partie texte et contiennent du code R. Ces codes peuvent être utilisés pour afficher des graphes, afficher des tableaux etc. Selon les paramètres de réglage des chunks, l'utilisateur peut décider d'afficher ou non les codes entrés, de les exécuter ou non, d'afficher ou non les avertissements lors de l'exécution des codes et bien d'autres.

NB : On peut aussi adapter notre document suivant la mise en forme d'un document source prédéfini.

3.3 Limite de R Markdown

R Markdown offre énormément de possibilités mais se trouve limité par la création des tableaux. En effet il est très difficile de créer des tableaux sur R Markdown et ceci en constitue une limite non négligeable.

4 R vers Excel (R2excel)

Il est possible de déployer des tableaux de R sur Excel et pour ce faire, l'on utilise principalement le package R2excel. Il permet de faire des sorties de tableaux de R sur Excel dans un format prédéfini. Il est utile dans le traitement et le rendu des résultats provenant de données d'enquête.

4.1 Autres packages

Les autres packages nécessaires pour faire du R vers Excel sont :

- *readxl* : permet d'importer des données depuis Excel
- *writexl* : permet d'exporter des données depuis R vers Excel
- *openxl*, *openxlsx* : permettent de lire, d'écrire et de manipuler des fichiers Excel dans R

4.2 Quelques fonctions

Quelques fonctions utiles pour faire du R vers Excel :

- *Write.xlsx()* : qui permet d'écrire un dataframe ou une liste de mots dans un fichier Excel
- *writeData()* : permet d'écrire des données dans un fichier Excel
- *write_xlsx_from_dataframes()* : permet d'écrire plusieurs dataframes dans un fichier Excel avec chaque dataframe dans une feuille de calcul séparée.
- *write_xlsx_from_lists()* : permet d'écrire plusieurs listes d'objets dans un fichier Excel.
- *write_xlsx_from_workbook()* : permet d'écrire un workbook existant dans un fichier Excel.
- *add_sheet()* : permet d'ajouter de nouvelles feuilles de calcul à un fichier Excel existant.
- *add_table()* : permet d'ajouter un tableau formaté à une feuille de calcul Excel.

5 R Quarto

Package très récent mis sur pied en janvier 2023, il a pour principal avantage de permettre à l'utilisateur de faire juste à l'aide de clics tout ce qu'on pouvait faire en utilisant les codes. L'objectif de *quarto* est de permettre de se passer d'office.

5.1 Réalisation d'un document *quarto*

Pour l'élaboration par défaut d'un document Quarto on peut procéder comme suit :

- *File* ensuite,
- *New file* puis on choisit soit :
  - *Quarto document* soit,
  - *Quarto présentation* selon ce qu'on veut faire.
- Dans la fenêtre qui s'ouvre entrez les informations nécessaires relatives à votre document quarto puis cliquez sur *create*

Avec quarto on peut insérer un titre, une image, une référence bibliographique, lien hypertexte, références bibliothèque et diagrammes. Les documents Quarto peuvent être produits soit sous format *Word* soit sous format *pdf* ou encore *html*.

La présentation avec Quarto peut se faire soit sur *PowerPoint*, *beamer* ou *Reveal.js* .

NB : Ne jamais mettre un lien à l'intérieur d'un rapport.

5.2 Quelques actions pouvant être réalisés avec R-Quarto et chemin d'accès à ses instructions

Action	Chemin d'accès
Insérer un titre	Aller dans l'onglet normal ensuite sur Header NB : on ne peut pas automatiser la numérotation des titres dans un document Word avec R-quarto. Aussi lors de l'affichage des titres l'indentation est faite de manière automatique.
Mise en gras	Peut être fait de trois façons : Bold ou format ensuite bold ou ctr+B
Insertion d'une note de bas de page	Insert ensuite cliquer sur foot note
Création d'une liste	Onglet format ensuite bullets & Numbering NB : ici l'utilisateur peut passer d'une liste non ordonnée à une liste ordonnée
Insertion d 'un tableau	Menu Table ensuite insert Table
Insertion d'une image	Menu insérer ensuite choisir Image/Figure : les images peuvent être insérer selon les emplacements de préférences de l'utilisateur et ceci peut être fait à l'aide de la fonction Layout. Log ratio oblige l'image de rester en ratio.
Insertion d'un saut de page	Ceci ne marche que sur les documents Word ou PDF et se fait à l'aide de {{< Page Break>}}
Création d'un livre	Projet ensuite New projet ensuite Quarto document et enfin Quarto Book
Insertion d'une formule mathématique	InlineMath insère une formule sur la même ligne que le texte précédent ; displayMath va à la ligne

5.3 Différence entre R-markdown et R-quarto

Sur R-markdown un tableau se fait de manière manuelle et c'est assez fastidieux contrairement à R-quarto où on le fait en quelques clics. Sur R-Markdown la réalisation d'un document se fait uniquement à l'aide des codes pourtant sur R-quarto on peut utiliser soit les codes soit les clics. Toutefois si on utilise uniquement les clics on peut avoir accès au code source ayant généré le document.

6 Textmining avec R

Le textmining ou encore analyse de texte est une approche consistant à transformer un texte non structuré en données structurées pour en faire des analyses statistiques. Il permet entre autres de faire l'analyse de sentiment. Il s'applique sur des bases de données ou sur des messages de conversations via les réseaux sociaux. Selon que l'on soit dans l'un ou l'autre des cas un certain nombre d'étapes sont à respecter.

6.1 Les marches à suivre

S'agissant des bases de données, il faut :

- Le prétraitement du texte qui est réalisé à l'aide des packages *tidytext*, *dplyr*, *tm*
- La détection des fréquences des mots ou des ngrammes à l'aide du package *ggplot*
- La construction de la matrice terme des mots avec le package *tm : :DocumentTermMatrix()*.
- Le nuage de mots qui se fait avec le package *wordcloud*
- Le réseau des mots : avec le package *ggraph*.

Pour ce qui est des messages de conversations, il faut réaliser ces étapes :

- La chronologie des messages est retracée avec le package *lubridate*
- Le classement des ID selon le nombre de messages envoyés (prétraitement)
- Les mots les plus fréquents utilisés par chaque ID
- La comparaison des mots les plus fréquents

6.2 Quelques fonctions utiles

Comme dit plus haut, les packages essentiellement utilisés pour le *textmining* sont *tidytext*, *dplyr* et *tm* . Mais à proprement parler le textmining se fait à l'aide des fonctions de ces packages dont nous présentons quelques unes :

- *tidytext : :unnest_tokens(tbl,output,input,token,format,to_lower)* : cette fonction permet de segmenter un texte en des mots individuels dans un tableau de données
- *tidytext : : stop_words()* : cette fonction permet de relever la liste des mots couramment utilisés tels que les articles (les mots sans grand intérêt)
- *dplyr : : anti_join()* : cette fonction permet de comparer deux tableaux de données et de renvoyer toutes les lignes du premier tableau qui n'ont pas correspondance dans le second tableau.
- *dplyr : : count()* : cette fonction permet de compter le nombre d'occurrences d'un mot dans un tableau de données
- *dplyr : : mutate()* : elle permet de créer de nouvelles colonnes ou de modifier des colonnes existantes
- *dplyr : : tibble()* : cette fonction permet de créer un tableau de données sur R de manière plus conviviale et cohérente
- *tm : : tm_map(txte1, removePunctuation)* : permet de supprimer les ponctuations.
  - ⋆ *tm : : tm_map(txte1, removeNumbers)* : permet de supprimer les nombres.

- ⋆ *tm : : tm_ map(txte1, removeWords, stopwords("english"))* : permet de supprimer les mots sans intérêt dans l'analyse
- ⋆ *tm : : tm_ map(txte1, stripWhitespace)* : permet de supprimer les espaces inutiles dans un corpus de texte
- ⋆ *tm : : tm_ map(txte1, tm_ reduce)* : permet de reduire les termes d'un corpus de texte
- ⋆ *tm : : VCorpus(VectorSource())* : permet de créer un objet de type VCorpus à partir d'un vecteur source.

7 Cartographie avec R

7.1 Définition

La cartographie est la représentation des données géographiques sur des cartes. Elle permet d'analyser des données géographiques en donnant une représentation graphique de celles-ci. Le langage R propose pour cela de nombreux packages qui permettent de faire des cartes de toutes sortes (des choroplèthes, les flux d'exportation par exemple) et de faire toutes sortes d'opérations sur les données géographiques comme le calcul des distances, les agrégations, les jointures etc.

7.2 Les données spatiales

Les données spatiales utilisées pour faire la cartographie se distinguent deux modes : le mode raster (ce sont des images comme des images satellites par exemple) et le mode vecteur qui utilise les points, les lignes, les polygones pour représenter les données géographiques. Les formats de données couramment utilisés incluent les fichiers shapefile, les données raster, les données GPS et les données provenant de services Web cartographiques. Les données externes, telles que les fonds de carte, les données topographiques, les données climatiques ou les images satellites, peuvent être intégrées dans les visualisations cartographiques en utilisant les fonctionnalités des packages R. Ces données peuvent être téléchargées à partir de sources en ligne ou de bases de données spécialisées, puis manipulées et superposées sur les cartes.

7.3 Quelques packages utiles

Plusieurs packages sont utilisés pour représenter les cartes avec R dont voici quelques-uns :

- *rgdal* : permet d'importer les données spatiales dans R.
- *ggplot2* : très utilisé pour les représentations graphiques dans R, ce package permet de combiner des données géographiques pour faire des cartes avec R et de les mettre en forme.
- *sp* : qui fournit la fonction *coodinates()* qui permet de convertir des objets de type data.frame en objet de type spatial.
- *sf* : qui utilise la fonction *st_ as_ sf()* pour convertir des objets de type data.frame en objet de type spatial. Il permet également d'effectuer de manipuler et d'analyser les données spatiales.
- *cartography* : offre des fonctionnalités avancées pour la création des cartes.
- *leaflet* : permet de créer des cartes interactives et dynamiques et offre des fonctionnalités telles que le zoom, le déplacement, les marqueurs, les infobulles, les filtres et les contrôles de sélection.

  Les cartes créées avec R peuvent être exportées sous différents formats, tels que des images (PNG, JPEG, SVG), des documents (PDF, HTML). Les applications Shiny peuvent également être utilisées pour partager des cartes interactives en ligne.

7.4 Limite de la cartographie sur R

Les mises en forme de carte avec R sont moins facile à établir qu'avec les logiciels de cartograhie comme ArcGIS. D'autres fonctionnalités spécifiques d'ArcGIS, telles que l'analyse d'accessibilité, les capacités de gestion des bases de données géospatiales, peuvent être plus adaptées à des cas d'utilisation spécifiques que ce qui est disponible dans les packages R.

8 Calcul parallèle avec R

Le calcul parallèle est la subdivision des tâches longues et fastidieuses à des parties de l'ordinateur non plus de manière automatique, mais à travers le bon vouloir de l'utilisateur. Les ordinateurs sont essentiellement des machines séquentielles et qui dit séquence dit suite d'instructions et fait référence à un ordre bien déterminé vu que les résultats de l'exécution d'une étape du processus précédent sont les entrées du processus suivant et ainsi de suite. Cette séquence gérée de manière automatique par l'ordinateur n'est pas une limite lorsque l'on traite des données non volumineuses c'est-à-dire de taille inférieure au nombre de lignes d'un fichier Excel ; mais l'est, notamment en ressources et en temps d'exécution, lorsqu'on traite des données volumineuses.

Faire du calcul parallèle revient donc à subdiviser soi-même les tâches que chaque composante de l'ordinateur (processeur) exécute. Il se définit aussi comme une subdivision d'une série de traitements en sous-opérations de manières indépendantes.

Pour pouvoir faire cette subdivision, le langage de programmation R met à notre disposition les packages *parallel* et *doparallel* à travers la programmation *MapReduce* (*Map* : partitionnement de la base de donnée en sous-bases de données et traitement des sous-bases ; *Reduce* : agrégation des résultats issus du Map) qui recense plusieurs fonctions dont :

- *detectcore(logical=FALSE)* : permet de détecter le nombre de cœurs physiques de la machine,
  - l'argument (logical=True) permet de détecter le nombre de cœurs de la machine y compris les processeurs logiques.

Le package *tictoc* permet d'évaluer le temps d'exécution d'une commande et permet de faire des comparaisons entre le temps de calcul du calcul parallèle et celui du calcul séquentiel.

NB :

Il est important de souligner que le package *fs* est d'une importance capitale pour l'importation des bases de données volumineuses vu que la fonction *readr()* de R fait éteindre la machine pour des big data.

Notons aussi que le calcul parallèle ne s'applique que sur des données volumineuses. Elle n'a pas d'intérêt sur des données non-volumineuses.

9 Utilisation de python dans R : Package reticulate

Pour utiliser python dans R on fait souvent usage de plusieurs packages à savoir *reticulate*, *rpython*, *pythonlnR*, *jupyter Notebooks*. Dans le cadre de cet exposé l'accent a été mis sur le package *reticulate*.

Le package reticulate est un package qui permet de mettre en réseau un environnement python et un environnement R. Il a été développé par trois auteurs : Kevin Ushey, JJ Allaire, Yuan Tang Publish Year : 2021.

9.1 Principe

Reticulate permet de quitter de l'environnement R et d'aller sur python puis de revenir sur R. il joue donc le rôle de passerelle entre les deux environnements.

Reticulate permet d'appeler python dans R de diverses façons que ce soit dans le R-Markdown ou le référencement de script python et permet également de convertir les objets python en objet R et objets R en objets python.

9.2 Installation du package reticulate

Les étapes à suivre pour l'installation de reticulate :

- Installer le package Reticulate sur R (install.package (reticulate))
- Avoir une version python au moins supérieur à 2.7 dans sa machine
- Configurer l'environnement python. Cet environnement peut être un environnement virtuel, un environnement conda ou un environnement python système.

Quand on est dans R et que on veut utiliser des objets python on peut utiliser l'instruction : ***objet = py$nom_objet_python***. De même si on n'est dans le code python et qu'on veut accéder à des objets de R on peut utiliser l'instruction ***objet = r.nom_objet_R***.

9.3 Recommandation

Pour utiliser python dans R il faut avoir la maitrise préalable des deux langages. Reticulate dépend du langage de programmation python : il faut toujours installer ce dernier avant de l'installer.

10 Équation non linéaire sur R

10.1 Définition

Un système d'équations non linéaires est un ensemble de plusieurs équations non linéaires simultanées. La résolution de ces équations présente beaucoup d'intérêt dans plusieurs domaines notamment en analyse de données où elle permet d'estimer les paramètres des modèles, d'ajuster les courbes et bien d'autres. Le langage de programmation R fournit des fonctions nécessaires pour résoudre ces systèmes.

10.2 Packages nécessaires

Les packages de R utilisés pour la résolution de systèmes d'équation non linéaires sont les suivants :

- rootSolve : ce package fournit des fonctions comme multiroot() pour la résolution de systèmes d'équations non linéaires. Il utilise des méthodes de résolution comme la méthode de Newton-Raphson
- nleqslv : ce package fournit également différentes fonctions pour la résolution de ces systèmes en proposant des méthodes basées sur le calcul de la jacobienne. Elle offre également la possibilité de choisir la méthode qu'on veut.
- stats : qui fournit la fonction optim() qui permet la résolution des systèmes non linéaires avec des méthodes indirectes (méthodes consistant à changer le système en un problème d'optimisation).

NB : D'une façon générale, les fonctions utilisées prennent en paramètre un vecteur de fonctions non linéaires, un vecteur de valeurs initiales et retournent des valeurs approchées des solutions du système.

11 R Shiny

Dans le monde de la statistique, il est important de créer des tableaux de bords qui permettent au statisticien de communiquer ses résultats de manière plus compréhensible à ses collaborateurs. Pour cela, le langage R propose le package Shiny qui sert à créer des applications web interactives permettant d'analyser des données, afficher des graphiques, exécuter les modèles statistiques de façon dynamique et tout cela sans avoir besoin de connaissances approfondies en HTML, CSS, Javascript.

11.1 Création d'un fichier R Shiny

Pour la création d'un fichier R Shiny, on peut procéder comme suit :

- *File* ensuite,
- *New file* puis,
- *Shiny Web Application* alors s'ouvre une fenêtre où on met le nom qu'on veut donner au dossier contenant le code de l'application. On peut aussi choisir de placer tout le code dans un seul fichier ou de le séparer en deux fichiers (fichier *ui* et fichier *server*). Après cela, on choisit l'emplacement du dossier créé puis on appuie *ok*. Il se créé alors un dossier qui porte le nom qu'on aura entré et qui contient soit un fichier soit deux fichiers d'extension .R (selon les choix qu'on aura faits) contenant un code par défaut.

Une fois que le code de l'application est terminé, on pourra cliquer sur le bouton *Run App* pour la lancer.

11.2 Les différentes parties d'une application *R Shiny*

- L'interface utilisateur : c'est la partie visible de l'application. Elle permet à l'utilisateur d'interagir avec celle-ci. Dans le code, on programme l'interface dans la partie « ui ». On peut par exemple y mettre une liste déroulante pour permettre à l'utilisateur d'entrer des données. On peut aussi y mettre des graphiques, des tableaux, des cartes, des champs de saisie et bien d'autres.
- Le server : c'est la partie où se gère les interactions avec l'utilisateur. On peut dire que c'est en quelques sortes « le cerveau » de l'application. On y programme donc comment l'application doit réagir suite à une entrée de l'utilisateur.

11.3 L'input et l'output

Le fonctionnement d'une application Shiny est basé sur le concept d'entrée et sortie (input et output).

Un input c'est tout ce que l'utilisateur entre dans l'application à partir de l'interface. Ce sont des éléments interactifs tels que des champs de saisie de texte, des boutons, des cases à cocher, des listes déroulantes, etc. Les inputs dans une application Shiny sont généralement définis dans la partie *ui* avec les fonctions spécifiques pour chaque type d'input comme *textInput*, *selectInput*, *checkboxInput* etc.

L'output affiche la réponse de l'application à l'input reçu. Les outputs peuvent être des zones de texte, des graphiques, des tableaux, des images. Ils sont définis dans la partie *ui* également à partir des fonctions comme *textOutput*, *plotOutput*, *tableOutput* etc. Les

calculs sont faits dans le *server* à partir des données fournies par les inputs et les résultats sont renvoyés aux outputs correspondants.

Shiny utilise des identifiants pour référencier chaque input et output. Tout cela est possible grâce à la réactivité de Shiny qui facilite la mise à jour dynamique des outputs en fonction des changements des inputs.

11.4 Quelques fonctions utilisées dans Shiny

Plusieurs fonctions son utilisées selon qu'on soit dans la partie *ui* ou dans la partie *server*.

- Fonctions utilisées dans l'ui : Ci-dessus nous avons cités quelques fonctions utilisées dans l'interface utilisateur à savoir *textInput* pour la création des champs de saisie de texte, *selectInput* pour les listes déroulantes, *tableOutput* pour créer un output tableau etc.
- Fonctions utilisées dans le server : Ce sont des fonctions qui permettent de renvoyer effectivement les résultats aux outputs. Il s'agit de *renderText* pour générer un texte, *renderPlot* pour générer un graphique, *renderTable* pour générer un tableau etc.

11.5 Limites de Shiny

Shiny est très limité pour le traitement de grands ensembles de données. On ne parle même pas de big data mais Shiny ne supporte pas les données de 50 000 lignes et plus. De plus lorsque des analyses statistiques intensives sont nécessaires, Shiny peut présenter des ralentissements ou des problèmes de temps de réponse.

12 Conclusion

Au terme de ce travail où il était question de faire la synthèse des différents thèmes d'exposés sur R à travers son environnement de développement intégré R-studio. Il en ressort que, le langage R, utilisé via R-Studio offre une multitude de fonctionnalités parmi lesquelles : l'analyse des données, la création des applications et des rapports dynamiques permettant la reproductivité avec shiny ; la manipulation des tableaux avec Gtsummary ; la production des beaux documents Word, PDF et html et des présentations avec beamer à l'aide de R-Quarto ou R-Markdown, le déploiement de python dans R avec reticulate.

Table des matières

1 Janitor **3**
1.1 Quelques fonctions de Janitor 3
1.2 Limites de janitor 3

2 gtsumary **4**
2.1 Quelques fonctions de gtsummary 4
2.2 Dépendance avec d'autres packages 4

3 R Markdown **5**
3.1 Création d'un fichier R Markdown 5
3.2 Les différentes parties d'un document *R Markdown* 5
3.3 Limite de R Markdown 5

4 R vers Excel (R2excel) **6**
4.1 Autres packages 6
4.2 Quelques fonctions 6

5 R Quarto **7**
5.1 Réalisation d'un document *quarto* 7
5.2 Quelques actions pouvant être réalisés avec R-Quarto et chemin d'accès à ses instructions 8
5.3 Différence entre R-markdown et R-quarto 8

6 Textmining avec R **9**
6.1 Les marches à suivre 9
6.2 Quelques fonctions utiles 9

7 Cartographie avec R **11**
7.1 Définition 11
7.2 Les données spatiales 11
7.3 Quelques packages utiles 11
7.4 Limite de la cartographie sur R 12

8 Calcul parallèle avec R **13**

9 Utilisation de python dans R : Package reticulate **14**
9.1 Principe 14
9.2 Installation du package reticulate 14
9.3 Recommandation 14

10 Équation non linéaire sur R **15**
10.1 Définition 15
10.2 Packages nécessaires 15

11 R Shiny **16**
11.1 Création d'un fichier R Shiny 16
11.2 Les différentes parties d'une application *R Shiny* 16
11.3 L'input et l'output 16
11.4 Quelques fonctions utilisées dans Shiny 17
11.5 Limites de Shiny 17

12 Conclusion **18**