EXAMEN FINAL SUR R

BENSALEM Akram

1er choix de package: DabR

Lien vers le github: https://github.com/CorentinBretonniere/CBRETONNIERE-PSBX/blob/main/dabr.pdf Par BRETONNIERE Corentin et SERREAU Antoine

Dans ce travail effectué, on nous présente un package R qui semble être un package pour la gestion de données d'un dataset. Au début, nous avons un tutoriel et plusieurs liens afin que nous puissions nous même installer la multitude de packages qui sont complémentaires de DabR comme dplyR par exemple. On nous incite même mariadb qui est un système de gestion de bases de données supportant des tables SQL. L'intsallation nécessite d'effectuer quelques commandes BASH pour les personnes possèdant unn système d'exploitation MAC-OS ou Linux.

Pour illustrer le code, une dataset a été importée depuis Kaggle. On a en premier lieu la fonction db-connect pour database connect, cette fonction nous permet de nous connecter à une base de données dans mariadb cela fonctionne à l'image d'un driver Impala par exemple. On peut par la suite vérifier que la connexion à la base de données grâce à la fonction "is.connected" qui renvoie un booléen en output sur le statut de connexion. La fonction "close_conn" est la complémentaire de "dbconnect" car à l'inverse de "dbconnect", cette fonction déconnecte de notre base de données dans mariadb. On a ensuite deux fonctions pour arranger et avoir une vision globale de notre base de données qui sont "delete" et "list_tables", ce sont des fonctions classiques qu'on peut retrouver dans tous les langages. Les fonctions "select_all" et "select" permettent d'effectuer des requêtes sur notre table comme sur SQL.

```
dabr::select(conn, "SELECT BPM FROM top50")
```

Finalement on va parler de la fonction "quote"" qui va tout simplement permettre de légender la table de données, le output étant un string.

Le travail sur ce package est bien structuré et comporte des lignes de code. Les étapes d'installation sont très limpides avec divers liens à notre disposition et en même temps peuvent initier des débutants à certaines commandes BASH. Une base de données a été importée rendant la présentation du package beaucoup plus visuelle. Je trouve le choix du package très pertinent car beaucoup de gens sont ammenés à gèrer et à trier une base de données, de plus comme certains modules de SparkR, ici on peut se servir à la fois de requêtes R mais également de requêtes SQL.

2nd choix de package: Package NetworkD3

Lien vers le github: https://github.com/clairemazzucato/PSBX/blob/main/Packages/NetworkD3/NetworkD3.pdf Par MAZZUCATO Claire et JUPITER Adrien

Ce package permettrait de créer des graphes, des dendogrammes et des arbres afin de représenter des réseaux d'où le "network" dans le nom du package. Au début nous avons une introduction à la notion de réseaux et pourquoi ce phénomène est étudié. Il existe d'autres packages sur R de modéliser les réseaux tel que Gephi par exemple. Une dataframe est même créée pour illustrer les fonctions de ce package, auquel on associe plusieurs autres packages comme dplyr ou tidyr. De plus on nous informe de l'existence de diverses représentations graphiques différentes pour les réseaux.

La représentation la plus simple d'un réseau est réalisée avec la fonction "simpleNetwork". Pour construire un graphe plus complexe on utilise la fonction "forceNetwork" qui donne en output un graphe avec des noeuds et liens où l'imbrication entre les différents noeuds est plus importante et/ou plus transverse. Le package permet également de réaliser des organigrammes comme l'organigramme du réseau de Sankey. Pour cela, la database brute est triée et on effectue une aggrégation avec la fonction "group_by", en l'occurence ici l'aggrégation a été faite par région. Il faut par la suite créer la table de noeuds puis de liens qui sont intégrés à la table grâce à la fonction "merge" qui réalise des joinutres entre plusieurs tables. Finalement, l'organigramme est généré avec la fonction sankeyNetwork avec en argument les noeuds et les liens bien-sûr. Finalement, la dernière fonctionnalité qui nous est démontrée dans la présentation de ce package est la modélisation d'un arbre de Reingold-Tilford avec la fonction "radialNetwork" à partir d'un dataset.

Le travail est bien structuré et contient plusieurs lignes de codes. On nous montre clairement quels packages installer pour pouvoir utiliser NetworkD3 de façon efficiente. Plusieurs dataframes ont été utilisées pour illustrer les fonctions du package, dont une qui a été créée par les étudiants eux-même. Les domaines d'application sont divers que ce soit pour l'étude de réseaux sociaux ou simplement l'imbrication entre différents acteurs. Je trouve le choix du sujet pertinent car avec le Big Data, comprendre les liens entre différentes entités peut-être complexe et grâce à ce package justement on peut modéliser les liens et les noeuds même s'ils sont nombreux et complexes.

3ème choix de package: Lubridate

Lien vers le github: https://github.com/GaspardPalay/PSBX/blob/main/TutorielLubridate/TutoLubridate.pdf Par PALAY Gaspard

Ce travail traite du package "lubridate". Ce package permettrait de gèrer des données de temps comme l'heure et des dates. Pour beaucoup de personnes le traitement de données comme des dates leur est problématique car souvent le langage R ou les autres langages comme Python reconnaissent des données brutes de date en tant que chaîne de caractères. D'où la manipulation restreinte de celle-ci. Au début de ce tutoriel, on nous montre comment installer le package puis on nous montre des exemples typiques de date et les fonctions qui conviennent pour les manipuler correctement.

Pour introduire justement la problématique visuellement, on a une première ligne de code avec la fonction "class" qui nous permet de voir commment une date écrite simplement peut-être reconnue par R. La première fonction du package qui nous est montré c'est la fonction "dmy" qui sont les acronymes pour day, month, year. Cette fonction permet de convertir une chaîne de caractères en type date suivant le format jour mois année. On pourrait également avec ce package convertir une chaîne de caractères en date-time, c'est-à-dire une date qui contiendrait des horaires. On se retrouve alors avec plusieurs variantes de la fonction "dmy" avec les focntions "ymd_hm","ymd_hms" ou tout simplement "hms" où les lettres h,m,s sont les acronymes respectives pour hour, minute, second. Il est même possible d'arrondir une date par sa dimension annuelle, mensuelle, journalière ou même horaire. On utilise la fonction "ceiling_date" pour arrondir vers le haut, "floor_date" pour arrondir vers le bas et "round_date" pour arrondir vers la valeur la plus proche. Il est possible grâce au package de calculer une durée qui s'écoule entre deux instants. On attribue deux instants distincts et on fait réaliser la soustraction entre les deux tout en les typant grâce aux fonctions "as_duration" et "as_period". La fonction "leap_year" quant à elle a la particularité de savoir si une année est bissextile. Le package prend en charge même des intervalles de temps tout en intégrant des fuseaux horaires à ces intervalles grâce à diverses fonctions et notamment celle qui est illustrée dans le code est la fonction "interval". Finalement on nous introduit le module POSIX qui dispose de plusieurs fonctions dont notamment "today" qui donne la date actuelle et "now" qui donne l'heure actuelle.

Le travail est bien structuré avec une introduction et des sections précises. La méthode d'installation est bien explicitée dans ce tutoriel. On a pas mal d'exemples de date et d'horaires, chaque exemple répond à une potentielle problématque concernant les dates sur le logiciel R. Je trouve le choix du sujet pertinent car il peut parler à un grand nombre de personnes qui justement on du mal à manipuler des dates dans des dataframes.

4ème choix de package: dplyr

Lien vers le github: https://github.com/gfontainepsb/Cours-R/blob/main/dplyr.pdf Par FONTAINE Grégoire

Le package étudié ici est "dplyr" est un package très utilisé pour manipuler des tables. Ses fonctions se caractérisent comme des verbes, ce qui rend ce package très intuitif. Dans ce tutoriel nous avons plusieurs lignes de code et de une dataframe qui a été importée. Il existe deux moyens de pouvoir installer ce package.

La première fonction qui nous est présentée est la fonction filter, comme nom l'indique cette fonction permet de filtrer la donnée qu'on a selon une ou plusieurs conditions. Ici on choisit que la colonne species devait prendre qu'une seule variable.

```
starwars %>%
  filter(species == "Droid")
```

La seconde fonction présentée est la fonction "select". Cette fonction est retrouvable sur tous les langages que ce soit sur python, SQL ou les HTML. Encore une fois son nom est explicite car cette fonction permet de sélectionner un ou plusieurs éléments depuis une liste ou une table. La fonction mutate quant à elle permet de créer une nouvelle variable/colonne qui dépend d'autres variables de la table. On pourrait faire le rapprochement avec la fonction "withColumn" dans PySpark. La dernière fonction est "arrange" qui existe également dans d'autres langages. Il permet de classer une liste ou une colonne dans un ordre. En l'occurence dans cet exemple l'auteur a choisi de trier par la colonne 'mass' par ordre croissant en introduisant "desc".

Le travail est bien structuré car dès le début on nous informe de l'utilité du package. Il comporte du code avec une importation de dataframe ce qui permet de rendre visuel ce que les fonctions effectuent. Le choix du package est pertinent car s'applique à toutes les tables et on sait que le travail de tri de la data est essentiel. Le tutoriel conviendrait à un débutant car il est concis et fait intervenir des fonctions intuitives et simples.

5ème choix de package: pdftools

Lien vers le github: https://github.com/chaymae-data/PSBX/blob/main/Extraire%20le%20contenu%20du%20PDF%20avec%20R.pdf Par GASMI Chaymae

Le package étduié ici est le package "pdftools". Ce package permettrait d'extraire du contenu qui est sous format PDF. En effet, on sait que le format PDF est difficile à manipuler car c'est difficile si on a pas de logiciel payant de changer son format. Ce que ce package permet d'extraire le text qui ensuite peut-être utilisé et traité.

La première étape est d'importer le fichier pdf avec la fonction "download.file". Le pdf peut être importé de manière locale ou via un lien url. Ensuite c'est là qu'on fait intervenir la fonction "pdf_text" qui justement va extraire toutes les chaînes de caractères du fichier. Il faut savoir que par défaut la fonction n'importe pas les espaces entre les chaînes de caractères. C'est pour ça que l'auteur du document a fait intervenir la fonction "strsplit" qui permet justement de diviser une chaîne de caractères et de laisser des espaces entre les mots.

Le travail est structuré et bien détailé. Il est concis et comporte du code. De plus un fichier pdf a été importé afin de confirmer le bon fonctionnement des fonctions. Le choix du travail est assez surprenant mais pertinent car je ne savais qu'il était possible de manipuler des objets tels que les pdf sur R. Ce package peut-être utile à des personnes qui n'arrivent pas à copier-coller du contenu depuis un PDF.

Auto-évaluation d'un tutoriel sur un package: gdata

Lien vers le github: https://github.com/AkramBensalemPSB/PSB/blob/main/package-gdata.pdf Par BENSALEM Akram

Si je devais évaluer le travail que j'ai effectué je dirais qu'il est plutôt bien structuré, il comporte une introduction qui explique les utilisations possibles de ce package dans les grandes lignes. Le tutoriel comporte plusieurs lignes de code pour diverses fonctions et une dataframe a été importée pour illustrer les fonctionnalités, d'ailleurs la dataframe est téléchargeable depuis mon github directement ou sur Kaggle. Le choix du travail

est pertinent car la panoplie de fonctions pour ce package est grande et peut-être utile à toute personne manipulant des tables.