TUTO DATA CLEANING D'UN DATAFRAME

Maxime & Lucas

CONTEXTE

Un fichier contenant des informations sur des personnes vous sont envoyées. Cependant ce fichier n'est pas traitable car les donnees ne sont pas correctement formatees. Votre but est donc de le nettoyer afin de pouvoir analyser les données.

NOTIONS ABORDEES

1.Importation de fichier csv 2.Data Cleaning 3.Méthodes et fonctions de bases sous R 4.Manipulation de dataframe

1. IMPORT FICHIER CSV

```
data = read.csv("C:/Users/allak/Desktop/PSB Cours/Mes cours/Maths pour le Big Data et programmation R/p
data
```

```
##   prenom                                email date_naissance          pays
## 1  Leila                     leila@example.com     23/01/1990        France
## 2 Samuel                 samuel_329@example.com     20/09/2001
## 3  Radia               choupipoune@supermail.eu  12 sept. 1984 Côte d'ivoire
## 4   Marc marco23@example.com, mc23@supermail.eu     10/02/1978        France
## 5   Heri                helloworld@supermail.eu     05/03/2008    Madagascar
## 6  Hanna                 hanna2019@supermail.eu     01/01/1970            24
## 7 samuël                 samuel_329@example.com                         Bénin
##   taille
## 1  1.49m
## 2  1.67m
## 3  153cm
## 4  1.65m
## 5  1.34m
## 6  3.45m
## 7  1.45m
```

```
str(data)#On affiche le type des differentes variables
```

```
## 'data.frame':    7 obs. of  5 variables:
##  $ prenom        : Factor w/ 7 levels "Hanna","Heri",..: 3 6 5 4 2 1 7
##  $ email         : Factor w/ 6 levels "choupipoune@supermail.eu",..: 4 6 1 5 3 2 6
##  $ date_naissance: Factor w/ 7 levels "","01/01/1970",..: 7 6 5 4 3 2 1
##  $ pays          : Factor w/ 6 levels "","24","Bénin",..: 5 1 4 5 6 2 3
##  $ taille        : Factor w/ 7 levels "1.34m","1.45m",..: 3 5 6 4 1 7 2
```

On voit que notre dataframe est brute et on cherche à nettoyer les données comprises dans la dataframe avant de faire l'analyse.

DATACLEANING - NETTOYAGE DE DONNEES

2. TRAITEMENT DE LA COLONNE “PRENOM”

Dans certaines dataframe, les noms ou prenoms peuvent etre en majuscule , en miniscules etc..voici la fonction qui permet d’avoir le meme format sur la colonne de prenom (Premiere lettre en majuscule et le reste en miniscule)

```
lower_case = function(value){
    print(paste('Voici la valeur que je traite:', value))
    return(tolower(value))
} #On crée une fonction ramenant la colonne prenom au meme format

data['prenom_min'] = apply(data['prenom'],1,lower_case)#puis on l'applique à nos prenoms de la datafram
```

```
## [1] "Voici la valeur que je traite: Leila"
## [1] "Voici la valeur que je traite: Samuel"
## [1] "Voici la valeur que je traite: Radia"
## [1] "Voici la valeur que je traite: Marc"
## [1] "Voici la valeur que je traite: Heri"
## [1] "Voici la valeur que je traite: Hanna"
## [1] "Voici la valeur que je traite: samuël"
```

```
data['prenom_min'] = NULL
data
```

```
##   prenom                                email date_naissance          pays
## 1  Leila                    leila@example.com     23/01/1990        France
## 2 Samuel                 samuel_329@example.com     20/09/2001              
## 3  Radia               choupipoune@supermail.eu  12 sept. 1984 Côte d'ivoire
## 4   Marc marco23@example.com, mc23@supermail.eu     10/02/1978        France
## 5   Heri                helloworld@supermail.eu     05/03/2008    Madagascar
## 6  Hanna                 hanna2019@supermail.eu     01/01/1970            24
## 7 samuël                 samuel_329@example.com                         Bénin
##   taille
## 1  1.49m
## 2  1.67m
## 3  153cm
## 4  1.65m
## 5  1.34m
## 6  3.45m
## 7  1.45m
```

3. TRAITEMENT COLONNE “EMAIL”

On se rend compte que certains clients possedent deux mails, on va chercher à en garder que le premier par la manipulation suivante :

```
#On declare la fonction qui permet de selectionner que les premiers mails
first = function(str){
    str = str[[1]]
    parts = strsplit(str,',')[[1]]
    first_part = parts[1]
    if(length(parts) >= 2)
```

```
        print(sprintf(' - Il y a plusieurs parties dans "%s", ne gardons que %s.',paste(parts,collapse=
    return(first_part)
}

#Application de la fonction a la colonne 'email'
data['email'] = apply(data['email'], 1, first)
```

```
## [1] " - Il y a plusieurs parties dans \"marco23@example.com mc23@supermail.eu\", ne gardons que marc
```

```
data
```

```
##   prenom                    email date_naissance          pays taille
## 1  Leila        leila@example.com     23/01/1990        France  1.49m
## 2 Samuel   samuel_329@example.com     20/09/2001                1.67m
## 3  Radia choupipoune@supermail.eu  12 sept. 1984 Côte d'ivoire  153cm
## 4   Marc      marco23@example.com     10/02/1978        France  1.65m
## 5   Heri  helloworld@supermail.eu     05/03/2008    Madagascar  1.34m
## 6  Hanna   hanna2019@supermail.eu     01/01/1970            24  3.45m
## 7 samuël   samuel_329@example.com                       Bénin  1.45m
```

4. TRAITEMENT COLONNE "DATE DE NAISSANCE"

Certaines dates de naissances sont au format numerique et d'autre format composé, on va essayer de les ramener tous au meme format.

```
data["date_naissance"] = as.Date(data$date_naissance , "%d/%m/%Y")
```

5. TRAITEMENT COLONNE "PAYS"

Dans la colonne pays on avait un chiffre comme nom d'un pays, ce qui est anormal et aussi une case vide, on va remplacer tous les noms non reconnus comme des pays par un NA

```
#On declare d'abord les noms des pays justes ou valables
VALID_COUNTRIES = c('France', "Côte d'ivoire", 'Madagascar', 'Bénin', 'Allemagne', 'USA')

#On declare la fonction qui permet de remplacer les noms non reconnus comme des pays par un NA
check_country = function(country){
    if(! country %in% VALID_COUNTRIES){
        print(sprintf(' - "%s" n\'est pas un pays valide, nous le supprimons.',country))
        return(NA)
    }
    return (country)
}

#Application de la fonction a la colonne pays
data['pays'] = apply(data['pays'], 1, check_country)
```

```
## [1] " - \"\" n'est pas un pays valide, nous le supprimons."
## [1] " - \"24\" n'est pas un pays valide, nous le supprimons."
```

```
data
```

```
##   prenom                    email date_naissance          pays taille
## 1  Leila        leila@example.com     1990-01-23        France  1.49m
## 2 Samuel   samuel_329@example.com     2001-09-20          <NA>  1.67m
## 3  Radia choupipoune@supermail.eu           <NA> Côte d'ivoire  153cm
## 4   Marc      marco23@example.com     1978-02-10        France  1.65m
## 5   Heri  helloworld@supermail.eu     2008-03-05    Madagascar  1.34m
## 6  Hanna   hanna2019@supermail.eu     1970-01-01          <NA>  3.45m
## 7 samuël   samuel_329@example.com           <NA>         Bénin  1.45m
```

6. TRAITEMENT DE LA TAILLE DES INDIVIDUS

On remarque dans notre dataframe que certains individus ont definis leur taille en cm et d'autres en m, pour avoir toute les tailles en m:

```
#une fonction permettant d'ignorer les valeurs en Cm et permettant de retirer aussi les unités 'm'
convert_height = function(height){
    found = regmatches(height, regexpr("[[:digit:]]\\.[[:digit:]]{2}m", height))
    if(length(found)==0){
        print(paste(height, ' n\'est pas au bon format. Il sera ignoré.'))
        return(NA)
    }else{
        value = substring(height,1,nchar(height)-1) # on enleve le dernier caractere, qui est 'm'
        return(as.numeric(value))
    }
}

#Une fonction permettant de remplacer une valeur anormale par la moyenne de la taille de tous les indiv
fill_height = function(height, replacement){
    if(is.na(height)){
        print(paste('Imputation par la moyenne :', replacement))
        return(replacement)
    }
    return(height)
}
#Application de notre fonction de conversion aux tailles des individus
data['taille'] = apply(data['taille'],1,convert_height)
```

```
## [1] "153cm  n'est pas au bon format. Il sera ignoré."
```

```
data['taille'] = apply(data['taille'], 1, function(t) if(!is.na(t) & t<3){t}else{NA})#Permet de d'affec

#Calcul de la moyenne et application de la fonction fill_height
mean_height = mean(as.numeric(data$taille), na.rm=TRUE)

#Une boucle qui permet de remplacer toutes les valeurs aberantes par la moyenne calculée ci-dessus
for(i in 1:nrow(data))
    data[i,'taille'] = fill_height(data[i,'taille'], mean_height)
```

```
## [1] "Imputation par la moyenne : 1.52"
## [1] "Imputation par la moyenne : 1.52"
```

7. COMPARAISON DE NOS DEUX DATAFRAME

Avant le traitement

```
old_data = read.csv("C:/Users/allak/Desktop/PSB Cours/Mes cours/Maths pour le Big Data et programmation
old_data
```

```
##   prenom                                email date_naissance          pays
## 1  Leila                     leila@example.com     23/01/1990        France
## 2 Samuel                  samuel_329@example.com     20/09/2001              
## 3  Radia               choupipoune@supermail.eu  12 sept. 1984 Côte d'ivoire
## 4   Marc marco23@example.com, mc23@supermail.eu     10/02/1978        France
## 5   Heri                helloworld@supermail.eu     05/03/2008    Madagascar
## 6  Hanna                 hanna2019@supermail.eu     01/01/1970            24
## 7 samuël                 samuel_329@example.com                       Bénin
##   taille
## 1  1.49m
## 2  1.67m
## 3  153cm
## 4  1.65m
## 5  1.34m
## 6  3.45m
## 7  1.45m
```

Apres le traitement

```
data
```

```
##   prenom                    email date_naissance          pays taille
## 1  Leila       leila@example.com     1990-01-23        France   1.49
## 2 Samuel   samuel_329@example.com     2001-09-20          <NA>   1.67
## 3  Radia choupipoune@supermail.eu           <NA> Côte d'ivoire   1.52
## 4   Marc      marco23@example.com     1978-02-10        France   1.65
## 5   Heri  helloworld@supermail.eu     2008-03-05    Madagascar   1.34
## 6  Hanna   hanna2019@supermail.eu     1970-01-01          <NA>   1.52
## 7 samuël   samuel_329@example.com           <NA>         Bénin   1.45
```