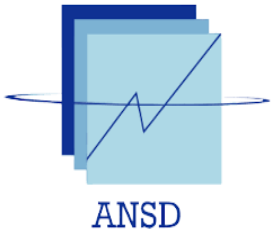
REPUBLIQUE DU SENEGAL

****

Un Peuple-Un But-Une Foi

****

Agence nationale de la statistique et de la démographie



École Nationale de la Statistique et de l'Analyse Economique Pierre Ndiaye

PROJET FINAL R

Présenté par :

Crépin MEDEHOUIN

Encadré par :

M. HEMA ABOUBACAR

2022 - 2023

Table des matières

RESUME

Ce projet vise à étudier les bioénergies durables pour les PME agroalimentaires en Afrique de l'Ouest en trois parties distinctes.

Dans la première partie, nous analyserons les bioénergies durables en utilisant un jeu de données de 250 observations et 33 variables du fichier "Base_Partie1.xlsx". L'objectif est d'obtenir des statistiques descriptives et de cartographier leur répartition géographique.

La deuxième partie traitera du nettoyage et de l'analyse des données avec un ensemble de données artificielles du fichier "Base_Partie2.xlsx". Nous préparerons les données et créerons des visualisations pour mieux comprendre les relations entre les variables.

Enfin, dans la troisième partie, nous développerons une application R Shiny interactive pour visualiser les événements politiques et les violences en Afrique de l'Ouest en utilisant la base de données "ACLED-Western_Africa.csv". L'application permettra aux utilisateurs de filtrer et de localiser ces événements sur une carte interactive.

En résumé, ce projet explore les opportunités de bioénergies durables pour les PME agroalimentaires en Afrique de l'Ouest grâce à des analyses statistiques, des visualisations et une application interactive.

```
#############################################################################################
#
# ------------------------ CHARGEMENT DES PACKAGES UTILISES----------------
#
#############################################################################################

# -------------------------MANIPULATION DE DONNEES :-----------------------------
library(dplyr)
library(tidyverse)
library(janitor)

# ------------------------VISUILISATION DE DONNEES :-------------------------------
library(ggplot2)
library(viridis)
library(ggnewscale)
library(ggspatial)
library(scales)
library(visdat)
library(ggpirate)#devtools::install_github("mikabr/ggpirate")


#-------------------------TABLEAU ET RESUMES DE DONNEES :---------------------------
library(gt)
library(flextable)
library(gtsummary)


# -------------------------LECTURE DE DONNEES :-------------------------------
library(readxl)


# ------------------------GESTION DE DONNEES SPATIALES :-------------------------
library(tmap)
library(raster)
library(leaflet)
library(sf)
```

```r
# ----------------------GESTION DE DISPOSITION DES GRAPHIQUES :--------------------------
library(gridExtra)



#
```

PARTIE I

PREPARATION DES DONNEES

1 DESCRIPTION

1.1 Importation et mise en forme

1.2 Importation de la base de donnée sous le nom “projet” et stockée dans un objet de type data.frame

```
#-------------------------------------------------------------------------------------------
# lecture du fichier Excel "Base_Partie 1.xlsx" :IMPORTATION
#-------------------------------------------------------------------------------------------

projet <- read_excel("Base_Partie 1.xlsx")

#
```

- Vérifiaction de l’id unique

```
# ------------------------------------------------------------------------------------------
#  RECHERCHE DE DOUBLONS PARMI LES IDENTIFIANT
#-------------------------------------------------------------------------------------------

projet %>%

  janitor::get_dupes(key)
```

```
## # A tibble: 0 x 34
## # i 34 variables: key <chr>, dupe_count <int>, q1 <chr>, q2 <chr>, q23 <chr>,
## #   q24 <dbl>, q24a_1 <dbl>, q24a_2 <dbl>, q24a_3 <dbl>, q24a_4 <dbl>,
## #   q24a_5 <dbl>, q24a_6 <dbl>, q24a_7 <dbl>, q24a_9 <dbl>, q24a_10 <dbl>,
## #   q25 <chr>, q26 <dbl>, q12 <chr>, q14b <chr>, q16 <chr>, q17 <chr>,
## #   q19 <chr>, q20 <chr>, filiere_1 <dbl>, filiere_2 <dbl>, filiere_3 <dbl>,
## #   filiere_4 <dbl>, q8 <chr>, q81 <chr>, gps_menlatitude <dbl>,
## #   gps_menlongitude <dbl>, submissiondate <dttm>, start <dttm>, ...
```

```
#
```

1.3 Résumer des valeurs manquantes par variable

```
# ----------------------------Étape 1: ----------------------------------------------------
#   Compter le nombre de valeurs manquantes dans chaque
#          colonne du dataframe "projet"
#-------------------------------------------------------------------------------------

projet_summary <- projet %>%

  summarise(across(everything(), ~sum(is.na(.))))



# ----------------------------Étape 2: ----------------------------------------------------
#   Transformer le dataframe en un format "long" avec
#     les noms de colonnes d'origine et le nombre
#           de valeurs manquantes
#-------------------------------------------------------------------------------------------

projet_long <- projet_summary %>%

  pivot_longer(everything(), names_to = "Variable", values_to = "Valeur manquante")
```

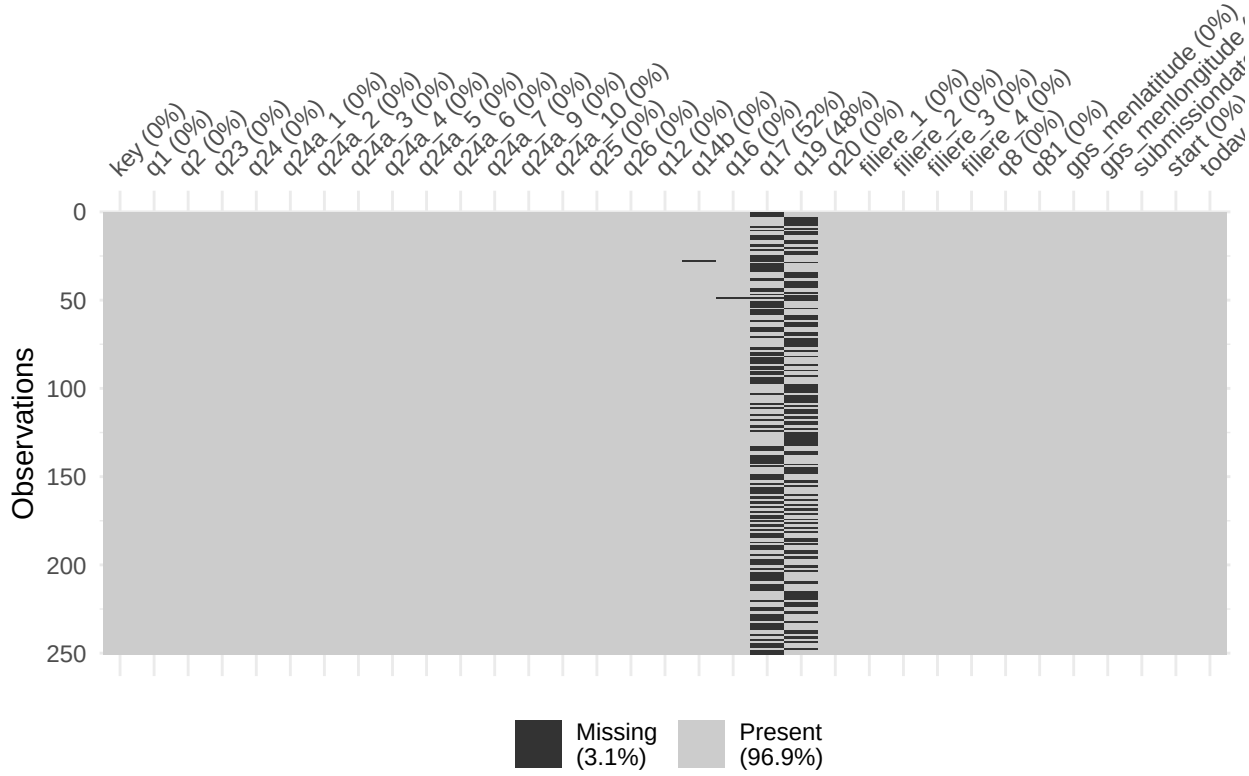
```r
# ----------------------------Étape 3: ----------------------------------------------------
#     Afficher le résultat sous forme d'un tableau
#       bien formaté en utilisant  "gt"
#-------------------------------------------------------------------------------------------

projet_long %>%

  gt(rowname_col = "Variable")
```

	Valeur manquante
key	0
q1	0
q2	0
q23	0
q24	0
q24a_1	0
q24a_2	0
q24a_3	0
q24a_4	0
q24a_5	0
q24a_6	0
q24a_7	0
q24a_9	0
q24a_10	0
q25	0
q26	0
q12	0
q14b	1
q16	1
q17	131
q19	120
q20	0
filiere_1	0
filiere_2	0
filiere_3	0

filiere_4	0
q8	0
q81	0
gps_menlatitude	0
gps_menlongitude	0
submissiondate	0
start	0
today	0

```
#
```

```
#-------------------------------------------------------------------------------------------
#       Utilisation de la fonction "vis_miss" pour
#         visualiser les données manquantes dans
#             le dataframe "projet"
#-------------------------------------------------------------------------------------------



visdat::vis_miss(projet)
```


key (0%)
q1 (0%)
q2 (0%)
q23 (0%)
q24 (0%)
q24a_1 (0%)
q24a_2 (0%)
q24a_3 (0%)
q24a_4 (0%)
q24a_5 (0%)
q24a_6 (0%)
q24a_7 (0%)
q24a_9 (0%)
q24a_10 (0%)
q25 (0%)
q26 (0%)
q12 (0%)
q14b (0%)
q16 (0%)
q17 (52%)
q19 (48%)
q20 (0%)
filiere_1 (0%)
filiere_2 (0%)
filiere_3 (0%)
filiere_4 (0%)
q8 (0%)
q81 (0%)
gps_menlatitude (0%)
gps_menlongitude
submissiondat
start (0%)
today
Observations
0
50
100
150
200
250
Missing
(3.1%)
Present
(96.9%)

```
#
```

1.4 Vérification des valeurs manquantes pour la variable key dans la base projet. Si oui, identifi la (ou les) PME concernée(s).

```
#-------------------------------------------------------------------------------------------
#       Filtrer les lignes où la colonne "key"
#     contient des valeurs manquantes (NA)
#-------------------------------------------------------------------------------------------

projet %>%

filter(is.na(key))
```

```
## # A tibble: 0 x 33
## # i 33 variables: key <chr>, q1 <chr>, q2 <chr>, q23 <chr>, q24 <dbl>,
## #   q24a_1 <dbl>, q24a_2 <dbl>, q24a_3 <dbl>, q24a_4 <dbl>, q24a_5 <dbl>,
## #   q24a_6 <dbl>, q24a_7 <dbl>, q24a_9 <dbl>, q24a_10 <dbl>, q25 <chr>,
## #   q26 <dbl>, q12 <chr>, q14b <chr>, q16 <chr>, q17 <chr>, q19 <chr>,
## #   q20 <chr>, filiere_1 <dbl>, filiere_2 <dbl>, filiere_3 <dbl>,
## #   filiere_4 <dbl>, q8 <chr>, q81 <chr>, gps_menlatitude <dbl>,
## #   gps_menlongitude <dbl>, submissiondate <dttm>, start <dttm>, ...
```

```
#
```

1.5 Création de variables

1.6 Nomination de quelques variables

```
#-------------------------------------------------------------------------------------------
#     La fonction rename() est utilisée pour renommer
#           les colonnes du dataframe.
```

```r
#-------------------------------------------------------------------------------------------

projet <- projet %>%

  rename(

    region = q1,

       departement = q2,

       sexe = q23
    )

#-------------------------------------------------------------------------------------------
#     Afficher les colonnes pour verification
#-------------------------------------------------------------------------------------------

projet %>%

  names()
```

```
##  [1] "key"              "region"           "departement"      "sexe"
##  [5] "q24"              "q24a_1"           "q24a_2"           "q24a_3"
##  [9] "q24a_4"           "q24a_5"           "q24a_6"           "q24a_7"
## [13] "q24a_9"           "q24a_10"          "q25"              "q26"
## [17] "q12"              "q14b"             "q16"              "q17"
## [21] "q19"              "q20"              "filiere_1"        "filiere_2"
## [25] "filiere_3"        "filiere_4"        "q8"               "q81"
## [29] "gps_menlatitude"  "gps_menlongitude" "submissiondate"   "start"
## [33] "today"
```

```r
#
```

1.7 Création de la variable sexe_2 qui vaut 1 si sexe égale à Femme et 0 sinon

```
#-------------------------------------------------------------------------------------------
#     la fonction "mutate()" du package dplyr pour ajouter
#         une nouvelle colonne nommée "sexe_2" avec
#   la condition ifelse(). Si la valeur de la colonne "sexe"
#     est égale à "Homme", alors la valeur de la colonne
#           "sexe_2" sera 0, sinon elle sera 1
#-------------------------------------------------------------------------------------------

projet <- projet %>%

  dplyr::mutate(sexe_2 = ifelse(sexe == "Homme", 0, 1), .after =
sexe)



#-------------------------------------------------------------------------------------------
#     Afficher les colonnes pour verification
#-------------------------------------------------------------------------------------------

projet %>%

  head()
```

```
## # A tibble: 6 x 34
##   key   region departement sexe  sexe_2   q24 q24a_1 q24a_2 q24a_3 q24a_4 q24a_5
##   <chr> <chr>  <chr>       <chr>  <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 uuid~ Diour~ Bambey      Femme      1    65      0      1      0      1      0
## 2 uuid~ Thiès  Mbour       Femme      1    52      1      1      0      0      1
## 3 uuid~ Thiès  Mbour       Femme      1    65      1      1      0      0      0
## 4 uuid~ Thiès  Mbour       Femme      1    38      1      1      0      0      1
## 5 uuid~ Zigui~ Bignona     Homme      0    40      1      1      1      0      0
## 6 uuid~ Zigui~ Oussouye    Femme      1    43      1      1      1      0      0
## # i 23 more variables: q24a_6 <dbl>, q24a_7 <dbl>, q24a_9 <dbl>, q24a_10 <dbl>,
```

```
## #   q25 <chr>, q26 <dbl>, q12 <chr>, q14b <chr>, q16 <chr>, q17 <chr>,
## #   q19 <chr>, q20 <chr>, filiere_1 <dbl>, filiere_2 <dbl>, filiere_3 <dbl>,
## #   filiere_4 <dbl>, q8 <chr>, q81 <chr>, gps_menlatitude <dbl>,
## #   gps_menlongitude <dbl>, submissiondate <dttm>, start <dttm>, today <dttm>
```

```
#
```

1.8 Création d'un data.frame nommé langues qui prend les variables key et les variables correspondantes

```
#-------------------------------------------------------------------------------------------
#     Selection a partie la fonction "select()" de dplyr :
#       la colonne "key" et toutes les variables commençant
#             par q24a_ "starts_with(q24a_)"
#-------------------------------------------------------------------------------------------

langues <- projet %>%

  dplyr::select(key,starts_with("q24a_"))



#-------------------------------------------------------------------------------------------
#     Afficher les colonnes pour verification
#-------------------------------------------------------------------------------------------

langues %>%

  head()
```

```
## # A tibble: 6 x 10
##   key            q24a_1 q24a_2 q24a_3 q24a_4 q24a_5 q24a_6 q24a_7 q24a_9 q24a_10
##   <chr>           <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>   <dbl>
## 1 uuid:68bff42b~      0      1      0      1      0      0      0      0       0
## 2 uuid:d70b3c7e~      1      1      0      0      1      0      0      0       0
```

```
## 3 uuid:0ac18b64~      1      1      0      0      0      0      0      0      0
## 4 uuid:c52cf5e4~      1      1      0      0      1      0      0      0      0
## 5 uuid:ac177870~      1      1      1      0      0      1      0      0      0
## 6 uuid:578097cf~      1      1      1      0      0      0      0      0      0
```

```
#
```

1.9 Création de la variable “parle” qui est égale au nombre de langue parlée par le dirigeant de la PME

```
# ------------------------------------------------------------------------------------
#    Sommer par ligne avec rowSums() les lignes donc
#      les langues prennent la valeur = 1 et selectionner
#          directement avec transmute() key et
#               la nouvellevariable
#-------------------------------------------------------------------------------------

nombre_langue <- projet %>%

  transmute(key,parle = rowSums(langues == "1") )



#-------------------------------------------------------------------------------------
#     Afficher les colonnes pour verification
#-------------------------------------------------------------------------------------

nombre_langue %>%

  head()%>%

  gt()
```

key	parle

uuid:68bff42b-1228-4c66-9bcc-e6d312d9fea6	2
uuid:d70b3c7e-3ca0-4358-bc59-3f7f6baf55e9	3
uuid:0ac18b64-7d85-4bb9-a842-698ac79909af	2
uuid:c52cf5e4-8c28-4e65-998b-3fe2a971a1a3	3
uuid:ac177870-001c-4ada-8747-c22ffe4e4596	4
uuid:578097cf-9af7-46e6-8992-d9079b14c342	3

```
#
```

1.10 Fusion de data.frame projet et langues

```
#-------------------------------------------------------------------------------------------
#   fusion des dataframes "projet" et "langues" en utilisant
#          la colonne "key" comme clé de fusion
#              avec la fonction  "merge()" de base en R
#-------------------------------------------------------------------------------------------

fusion = merge(projet,langues, by = "key")


#
```

1.11 Analyses descriptives

1.12 La répartion des PME

```
#-------------------------------------------------------------------------------------------
#         Personnaliser les paramètres de langue pour
#   le français (fr) dans les tables générées par gtsummary
#-------------------------------------------------------------------------------------------



theme_gtsummary_language("fr",decimal.mark = ",", big.mark =" ")




#-------------------------------------------------------------------------------------------
#  La fonction tbl_summary() du package gtsummary
#   utilisée pour créer le tableau de résumé des colonnes
#     "SEXE" , "NIVEAU D'INSTRUCTION","STATUT JURIDIQUE"
#        et "PROPRIETAIRE/LOCATAIRE"
#-------------------------------------------------------------------------------------------



tab1 = projet %>%


  gtsummary::tbl_summary(

    include = c("sexe","q25","q12","q81"),
```

```
    statistic =  list( all_categorical() ~ "{p}% ({n}/{N})"),

    label = list( q25 ~ "NIVEAU D'INSTRUCTION",

                  q12 ~ "STATUT JURIDIQUE",

                  q81 ~ "PROPRIETAIRE/LOCATAIRE",

                  sexe ~ "SEXE"
                  )
  )


#-------------------------------------------------------------------------------------------
#   Ajout de l'option supplémentaire by = sexe pour grouper
#       les statistiques par la variable "sexe" : croisement
#-------------------------------------------------------------------------------------------



tab2 = projet %>%


  gtsummary::tbl_summary(

    include = c("sexe","q25","q12","q81"),

    by = sexe,

    statistic =  list( all_categorical() ~ "{p}% ({n}/{N})"),

    label = list( q25 ~ "NIVEAU D'INSTRUCTION",

                  q12 ~ "STATUT JURIDIQUE",
```

```
                  q81 ~ "PROPRIETAIRE/LOCATAIRE")
  )



# ------------------------------------------------------------------------------------------
# la fonction tbl_merge() du package gtsummary pour fusionner
#  les deux tableaux de résumé tab1 et tab2 créés précédemment
#-------------------------------------------------------------------------------------------



tbl_merge(list(tab1,tab2),tab_spanner = c("ANALYSE UNIVARIE","ANALYSE BIVARIE"))%>%


  bold_labels() %>%


  italicize_levels()  %>%


  modify_header(
    update = list( label ~ "**VARIABLE**",
                   all_stat_cols(stat_0 = FALSE) ~ "**{level}** (n={n}, {style_percent(p)}%)"
  )) %>%


  as_flex_table() %>%


  fontsize(size = 8) %>%


  width(width = 1.65)
```

	ANALYSE UNIVARIE	ANALYSE BIVARIE	
VARIABLE	**N = 250**[1]	**Femme** (n=191, 76%)[1]	**Homme** (n=59, 24%)[1]
SEXE			
Femme	76% (191/250)		
Homme	24% (59/250)		
NIVEAU D'INSTRUCTION			
Aucun niveau	32% (79/250)	37% (70/191)	15% (9/59)
Niveau primaire	22% (56/250)	25% (48/191)	14% (8/59)
Niveau secondaire	30% (74/250)	29% (56/191)	31% (18/59)
Niveau Superieur	16% (41/250)	8,9% (17/191)	41% (24/59)
STATUT JURIDIQUE			
Association	2,4% (6/250)	1,6% (3/191)	5,1% (3/59)
GIE	72% (179/250)	78% (149/191)	51% (30/59)
Informel	15% (38/250)	17% (32/191)	10% (6/59)
SA	2,8% (7/250)	0,5% (1/191)	10% (6/59)
SARL	5,2% (13/250)	1,0% (2/191)	19% (11/59)
SUARL	2,8% (7/250)	2,1% (4/191)	5,1% (3/59)
PROPRIETAIRE/LOCATAIRE			
Locataire	9,6% (24/250)	8,4% (16/191)	14% (8/59)
Propriétaire	90% (226/250)	92% (175/191)	86% (51/59)

[1]% (n/N)

1.13 les statistiques descriptives de votre choix sur les autres variables

```
#-------------------------------------------------------------------------------------------
#         Personnaliser les paramètres de langue pour
#   le français (fr) dans les tables générées par gtsummary
#-------------------------------------------------------------------------------------------


theme_gtsummary_language("fr",decimal.mark = ",", big.mark =" ")

#-------------------------------------------------------------------------------------------
#  La fonction subset(filiere_1 == 1) pour créer un
# sous-ensemble T1 du dataframe projet, en ne conservant que
#     les lignes où la valeur de la colonne "filiere_1"
#-------------------------------------------------------------------------------------------
#   rename(Arachide = filiere_1): Cela renomme la colonne
#  "filiere_1" en "Arachide" dans le nouveau dataframe
#
#-------------------------------------------------------------------------------------------
#   La fonction tbl_summary() du package gtsummary pour
#   créer un tableau de résumé avec les variables "region"
#            et "Arachide"
#-------------------------------------------------------------------------------------------




T1 = projet %>%

  subset(filiere_1 == 1) %>%

  rename(
```

```
    Arachide = filiere_1

  ) %>%

  gtsummary::tbl_summary(

  include = c("region","Arachide"),

  by = Arachide,

  statistic = list( all_categorical() ~ "{p}% ({n}/{N})"),

  label = region ~ "REGION"

)




#-------------------------------------------------------------------------------------------
#     Meme proceder pour la filiere anacarde
#-------------------------------------------------------------------------------------------



T2 = projet %>%

  subset(filiere_2 == 1) %>%

  rename(

    Anacarde = filiere_2

  ) %>%

  gtsummary::tbl_summary(
```

```
  include = c("region","Anacarde"),

  by = Anacarde,

  statistic = list( all_categorical() ~ "{p}% ({n}/{N})"),

  label = region ~ "REGION"

)
```

```
#-------------------------------------------------------------------------------------------
#     Meme proceder pour la filiere mangue
#-------------------------------------------------------------------------------------------
```

```
T3 = projet %>%

  subset(filiere_3 == 1) %>%

  rename(

    Mangue = filiere_3

  ) %>%

  gtsummary::tbl_summary(

    include = c("region","Mangue"),
```

```
  by = Mangue,

  statistic = list( all_categorical() ~ "{p}% ({n}/{N})"),

  label = region ~ "REGION"

)
```

```
#-------------------------------------------------------------------------------------------------
#     Meme proceder pour la filiere riz
#-------------------------------------------------------------------------------------------------
```

```
T4 = projet %>%

  subset(filiere_4 == 1) %>%

  rename(

    Riz = filiere_4

  ) %>%

  gtsummary::tbl_summary(

    include = c("region","Riz"),

    by = Riz,
```

```
    statistic = list( all_categorical() ~ "{p}% ({n}/{N})"),

    label = region ~ "REGION"

)
```

```
# ------------------------------------------------------------------------------------
# la fonction tbl_merge() du package gtsummary pour fusionner
#  les quatres tableaux de résumé T1, T2, T3 et T4 créés précédemment
#-------------------------------------------------------------------------------------
```

```
tbl_merge(list(T1,T2,T3,T4))%>%

  bold_labels() %>%

  italicize_levels() %>%

  modify_header(

  update = list(

    label ~ "**FILIERES**",

    all_stat_cols(stat_0 = FALSE) ~ "**{level}** (n={n},

    {style_percent(p)}%)"
)) %>%
```

```
modify_spanning_header(all_stat_cols() ~ "**NOMBRE DE DIRIGEANT / PROPRIETAIRE PAR FILIERE**") %>%

as_flex_table() %>%

fontsize(size = 8) %>%

width(width = 1.3)
```

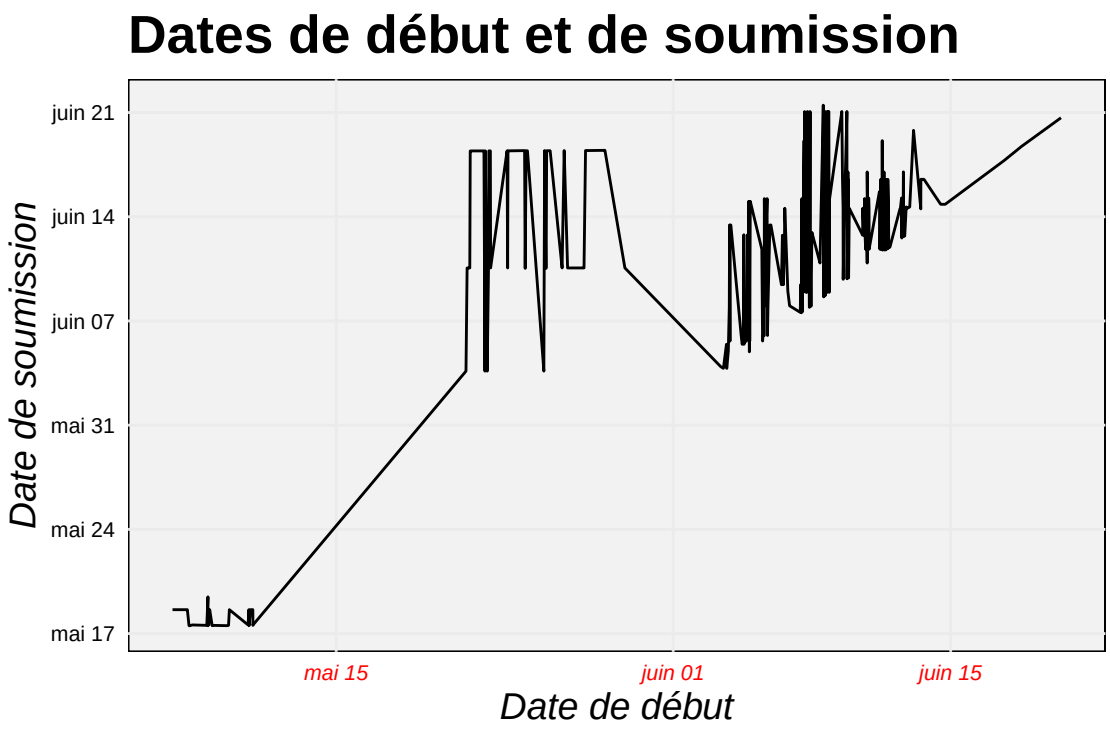
	NOMBRE DE DIRIGEANT / PROPRIETAIRE PAR FILIERE			
FILIERES	**1** (n=108, 100%)[1]	**1** (n=61, 100%)[1]	**1** (n=89, 100%)[1]	**1** (n=92, 100%)[1]
REGION				
Diourbel	31% (33/108)		1,1% (1/89)	
Fatick	11% (12/108)	34% (21/61)	3,4% (3/89)	4,3% (4/92)
Kaffrine	7,4% (8/108)		5,6% (5/89)	1,1% (1/92)
Kaolack	19% (20/108)		7,9% (7/89)	4,3% (4/92)
Kolda	0,9% (1/108)	8,2% (5/61)		4,3% (4/92)
Saint-Louis	0,9% (1/108)		47% (42/89)	
Thiès	25% (27/108)		28% (25/89)	35% (32/92)
Ziguinchor	5,6% (6/108)	51% (31/61)	6,7% (6/89)	47% (43/92)
Dakar		1,6% (1/61)		1,1% (1/92)
Sédhiou		4,9% (3/61)		3,3% (3/92)

[1]% (n/N)

```
#
```

```r
#-----------------------------------------------------------------------------------------------
#  Le graphique représente les dates de début (start)
#     en fonction des dates de soumission (submissiondate)
#             avec ggplot2.
#-----------------------------------------------------------------------------------------------
# La fonction geom_line() pour relier les points de données
# par des lignes pour créer un graphique linéaire
#-----------------------------------------------------------------------------------------------

ggplot(projet, aes(x = start, y = submissiondate)) +

  geom_line() +

  labs(title = "Dates de début et de soumission",

       x = "Date de début",

       y = "Date de soumission") +

  theme_minimal() +

  theme(

    plot.title = element_text(color = "black", size = 20, face = "bold"),

    axis.text.x = element_text(color = "red", size = 8, face = "italic"),

    axis.text.y = element_text(color = "black", size = 8),

    axis.title.x = element_text(color = "black", size = 14, face = "italic"),

    axis.title.y = element_text(color = "black", size = 14, face = "italic"),

    panel.grid.minor = element_blank(),

    panel.background = element_rect(fill = "#F2F2F2"),
```

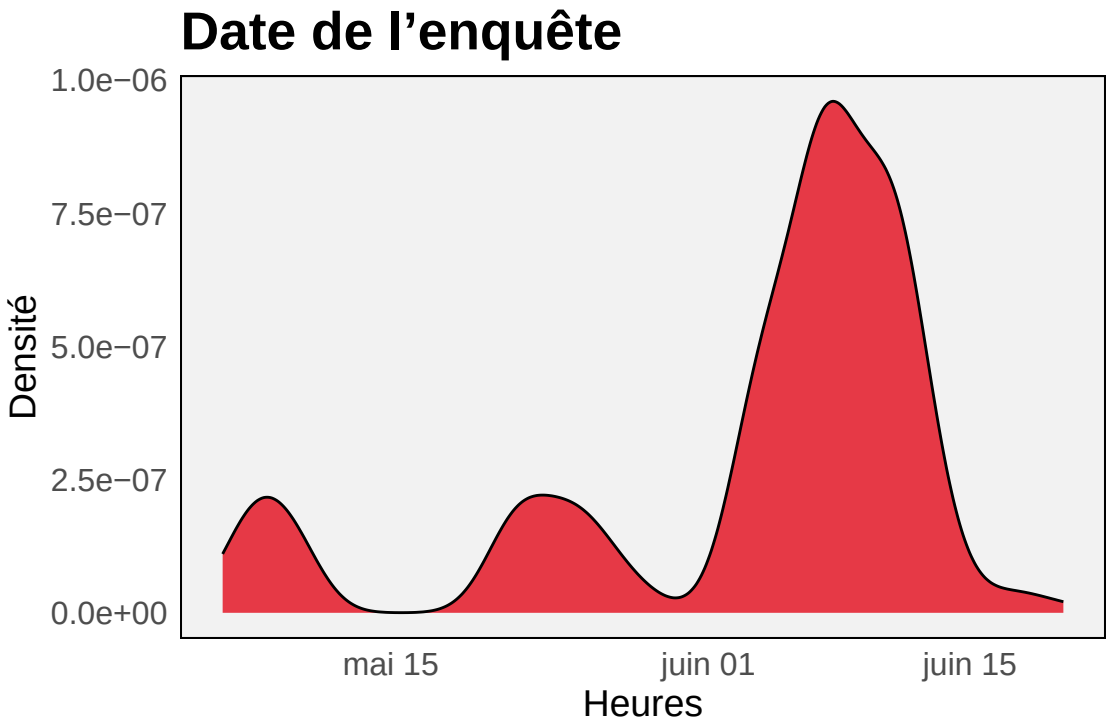
```
    legend.background = element_rect(fill = "#F2F2F2", color = "black"),

    plot.margin = margin(1, 1, 1, 1, "cm")
  )
```




```
#
```

```r
#-----------------------------------------------------------------------------------------
#  Le graphique de densité avec geom_density représentant
#    la distribution des valeurs de la variable today
#-----------------------------------------------------------------------------------------
# Un peu d'esthetique avec la fonction theme ()
#-----------------------------------------------------------------------------------------

ggplot(projet) +

  aes(x = today) +

  geom_density(fill = "#e63946", col = "black", show.legend = FALSE) +

  ggtitle("Date de l'enquête") +

  xlab("Heures") +

  ylab("Densité") +

  theme_minimal() +

  theme(

    plot.title = element_text(color = "black", size = 20, face = "bold"),

    axis.text.x = element_text(size = 12),

    axis.text.y = element_text(size = 12),

    axis.title.x = element_text(size = 14),

    axis.title.y = element_text(size = 14),

    panel.grid.major = element_blank(),

    panel.grid.minor = element_blank(),
```

```
  panel.background = element_rect(fill = "#F2F2F2"),

  legend.position = "bottom",

  legend.title = element_text(size = 12),

  legend.text = element_text(size = 10),

  legend.background = element_rect(fill = "white", color = "black"),

  plot.margin = margin(1, 1, 1, 1, "cm")
)
```



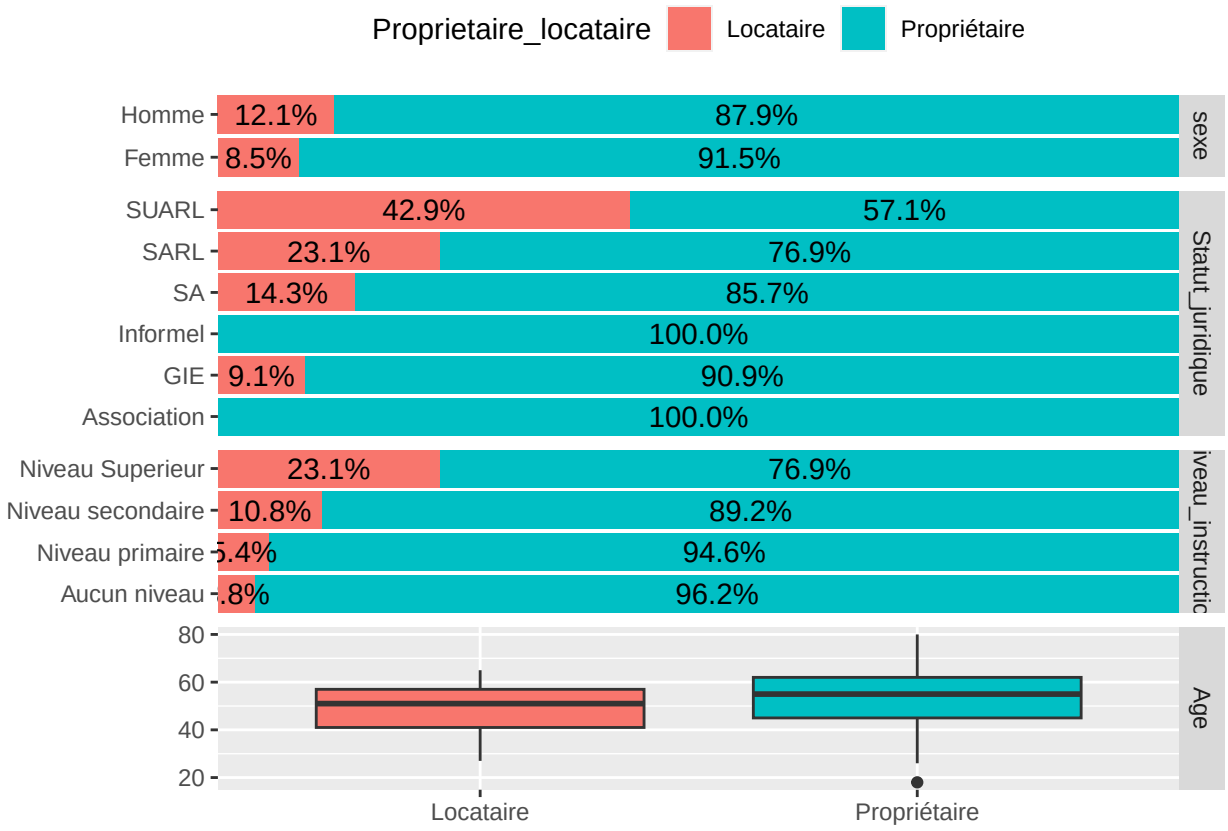

```
#
```

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
#-----------------------------------------------------------------------------------------------
# Creer une nouvelle variable Age
#-----------------------------------------------------------------------------------------------
projet$Age  =  projet$q24

#-----------------------------------------------------------------------------------------------
# Filtrer les age inferieur a 120 pour etre realiste
#-----------------------------------------------------------------------------------------------

Resume  =  projet %>%

  filter(

    Age < 120

    ) %>%

  rename(

    Proprietaire_locataire = q81,Statut_juridique  = q12,

    Niveau_instruction = q25

  )

#-----------------------------------------------------------------------------------------------
# Un exemple de resumer  des variable avec  ggbivariate()
#-----------------------------------------------------------------------------------------------

ggbivariate(
```

```
data = Resume,

          outcome = "Proprietaire_locataire",

          explanatory = c("sexe", "Statut_juridique",

                          "Niveau_instruction", "Age"

                          )

          )
```




```
#
```

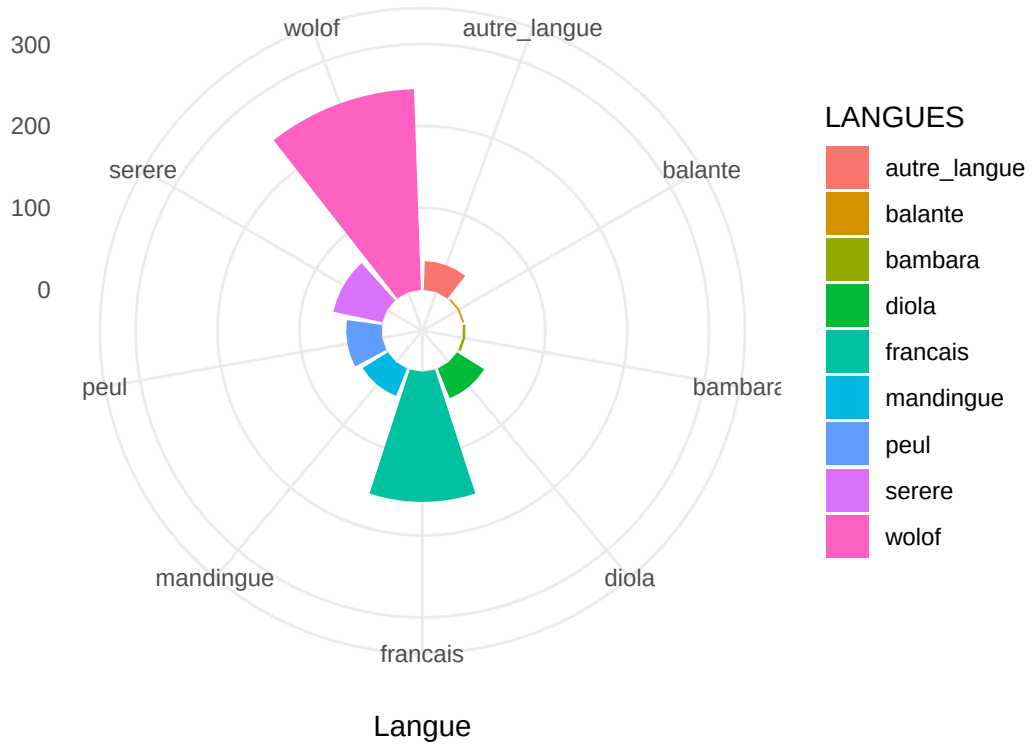
```r
#-------------------------------------------------------------------------------------------
#               Selection des langues et renommer
#-------------------------------------------------------------------------------------------
vect_langue = c("francais","wolof",
                "diola","serere","peul",
                "mandingue","balante","bambara",
                "autre_langue")



data <- projet %>%

  dplyr::select(starts_with("q24a_"))

names(data) = vect_langue



#-------------------------------------------------------------------------------------------
#  Regrouper par langue en data1 ..... data9
#      i.e  Sommer le nombre de langue parler
#-------------------------------------------------------------------------------------------

data1 =  data %>%
filter(francais == 1) %>%
  dplyr::mutate(francais = ifelse(francais == 1, "francais"))%>%
  group_by(francais)%>%
  count(name = "total_langue")

data2 =  data %>%
filter(wolof == 1) %>%
  dplyr::mutate(wolof = ifelse(wolof == 1, "wolof"))%>%
  group_by(wolof)%>%
  count(name = "total_langue")

data3 =  data %>%
filter(diola == 1) %>%
```

```
  dplyr::mutate(diola = ifelse(diola == 1, "diola"))%>%
  group_by(diola)%>%
  count(name = "total_langue")

data4 =  data %>%
filter(serere == 1) %>%
  dplyr::mutate(serere = ifelse(serere == 1, "serere"))%>%
  group_by(serere)%>%
  count(name = "total_langue")

data5 =  data %>%
filter(peul  == 1) %>%
  dplyr::mutate(peul  = ifelse(peul  == 1, "peul"))%>%
  group_by(peul )%>%
  count(name = "total_langue")

data6 =  data %>%
filter(mandingue  == 1) %>%
  dplyr::mutate(mandingue  = ifelse(mandingue  == 1, "mandingue"))%>%
  group_by(mandingue)%>%
  count(name = "total_langue")



data7 =  data %>%
filter(balante  == 1) %>%
  dplyr::mutate(balante  = ifelse(balante == 1, "balante"))%>%
  group_by(balante)%>%
  count(name = "total_langue")

data8 =  data %>%
filter(bambara  == 1) %>%
  dplyr::mutate(bambara  = ifelse(bambara == 1, "bambara"))%>%
  group_by(bambara)%>%
  count(name = "total_langue")

data9 =  data %>%
```

```r
filter(autre_langue  == 1) %>%
  dplyr::mutate(autre_langue  = ifelse(autre_langue == 1, "autre_langue"))%>%
  group_by(autre_langue)%>%
  count(name = "total_langue")



#-------------------------------------------------------------------------------------------------
#    Fusionner les 10 data
#-------------------------------------------------------------------------------------------------

LANGUE <- bind_rows(

data1 %>%
  rename(LANGUES = francais),

data2 %>%
  rename(LANGUES = wolof),

data3 %>%
  rename(LANGUES = diola),

data4 %>%
  rename(LANGUES = serere),

data5 %>%
  rename(LANGUES = peul),

data6 %>%
  rename(LANGUES = mandingue),

data7 %>%
  rename(LANGUES = balante),

data8 %>%
  rename(LANGUES = bambara),
```

```r
data9 %>%
  rename(LANGUES = autre_langue)
)



#-------------------------------------------------------------------------------------------------
#  Créer un graphique à barres à partir du dataframe avec geom_bar
#-------------------------------------------------------------------------------------------------

ggplot( data = LANGUE , aes( x= LANGUES, y = total_langue, fill = LANGUES ))+

geom_bar( stat = "identity")+

  labs(title = "GRAPHIQUE DES LANGUES PARLEES",
       x = "Langue"  ,
       y ="")+

theme_minimal()+

coord_polar(start = 0)+

ylim( -50, 300)+
  theme(
    plot.title = element_text(color = "black", size = 20, face = "bold")
  )
```

GRAPHIQUE DES LANGUES PARLEES

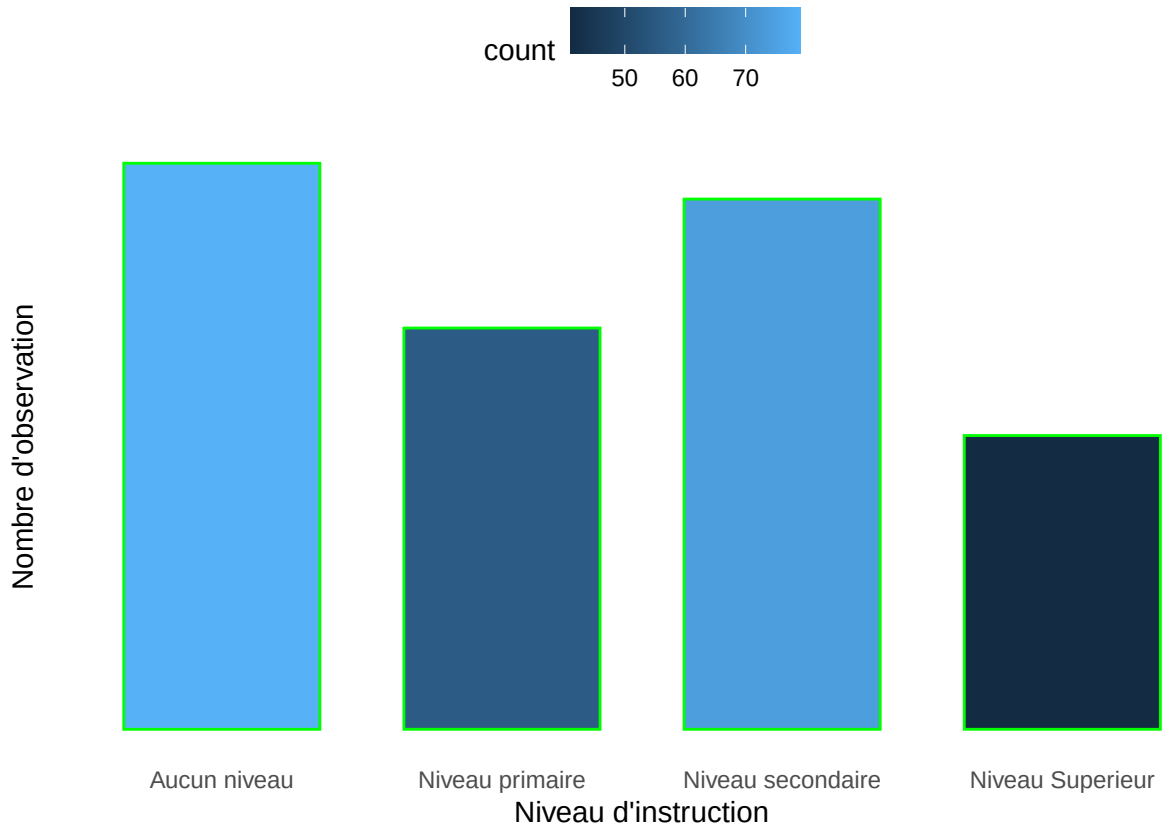



```
#
```

```r
#-----------------------------------------------------------------------------------------
#  Créer un graphique à barres à partir du dataframe avec geom_bar
#-----------------------------------------------------------------------------------------


Instruction = projet %>%

  rename(Niveau_instruction = q25)


ggplot(data = Instruction )+

  geom_bar(aes( x =Niveau_instruction,
                y = ..prop..,group = 1 ,
                fill = ..count..),
           col = "green", width = 0.7)+

  labs(x= "Niveau d'instruction", y = "Nombre d'observation")+

  theme_minimal() +

  theme(
    panel.grid = element_blank(),

    panel.border = element_blank(),

    axis.text.y = element_blank(),

    axis.ticks = element_blank(),

    legend.position = "top"
  )
```

```
## Warning: The dot-dot notation (‘..prop..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(prop)‘ instead.
## This warning is displayed once every 8 hours.
```

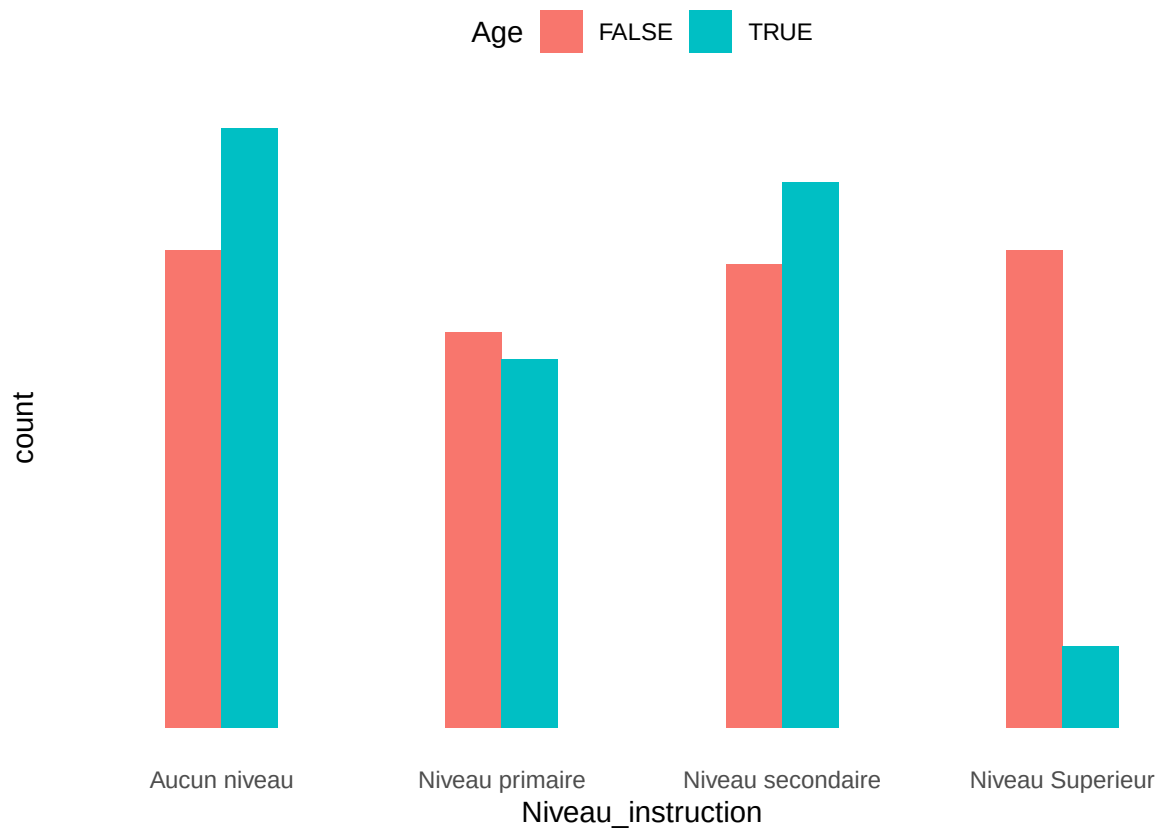
```
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```




```
#-------------------------------------------------------------------------------
```

```
Instruction = projet %>%

  rename(Niveau_instruction = q25)


#-------------------------------------------------------------------------------

Instruction$Age <- Instruction$q24 > median(Instruction$q24)

#-------------------------------------------------------------------------------

#-------------------------------------------------------------------------------
#  Créer un graphique à barres à partir du dataframe avec geom_bar
```

```r
#-----------------------------------------------------------------------------------------

ggplot(data = Instruction) +

  geom_bar(aes(x = Niveau_instruction, fill = Age),

           position = "dodge", width = 0.4) +

  theme_minimal() +

  theme(
    panel.grid = element_blank(),

    panel.border = element_blank(),

    axis.text.y = element_blank(),

    axis.ticks = element_blank(),

    legend.position = "top"
  )
```




```
#-------------------------------------------------------------------------------------------
#     En fonction du niveau d'instruction, nous affichons ceux qui ont un age superieur a la mediane
#      et ceux inferieur a la medianne . Alors on a vrai
#           pour ceux superieur a la moyenne et faux sinon
#-------------------------------------------------------------------------------------------
```

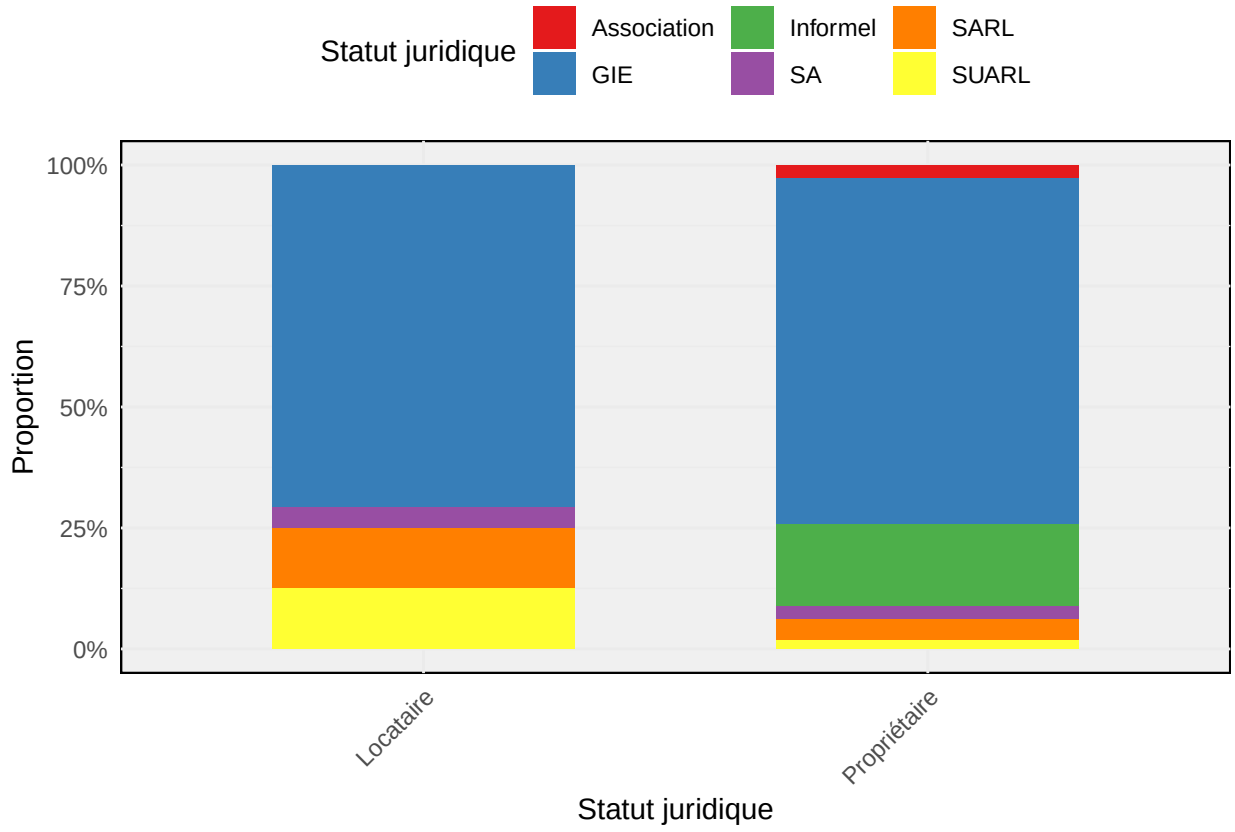
```
#-------------------------------------------------------------

Statut <- projet %>%

  rename(Statut_juridique = q12,

                         Proprietaire_locataire = q81)

# ------------------------------------------------------------------------------------------
#   Calculer la proportion des différents statuts juridiques
#     pour chaque catégorie de propriétaire/locataire
```

```r
#-------------------------------------------------------------------------------------------

prop_data <- Statut %>%

  group_by(Proprietaire_locataire, Statut_juridique) %>%

  summarize(Proportion = n()) %>%

  group_by(Proprietaire_locataire) %>%

  mutate(Proportion = Proportion / sum(Proportion))
```

```
## `summarise()` has grouped output by 'Proprietaire_locataire'. You can override
## using the `.groups` argument.
```

```r
#-------------------------------------------------------------------------------------------
# Création du graphique à barres empilées
#-------------------------------------------------------------------------------------------

ggplot(data = prop_data, aes(x = Proprietaire_locataire, y = Proportion, fill = Statut_juridique)) +

  geom_bar(stat = "identity", width = 0.6) +

  xlab("Statut juridique") +

  ylab("Proportion") +

  labs(fill = "Statut juridique") +

  scale_y_continuous(labels = scales::percent) +

  scale_fill_brewer(palette = "Set1") +

  theme_minimal() +

  theme(
```

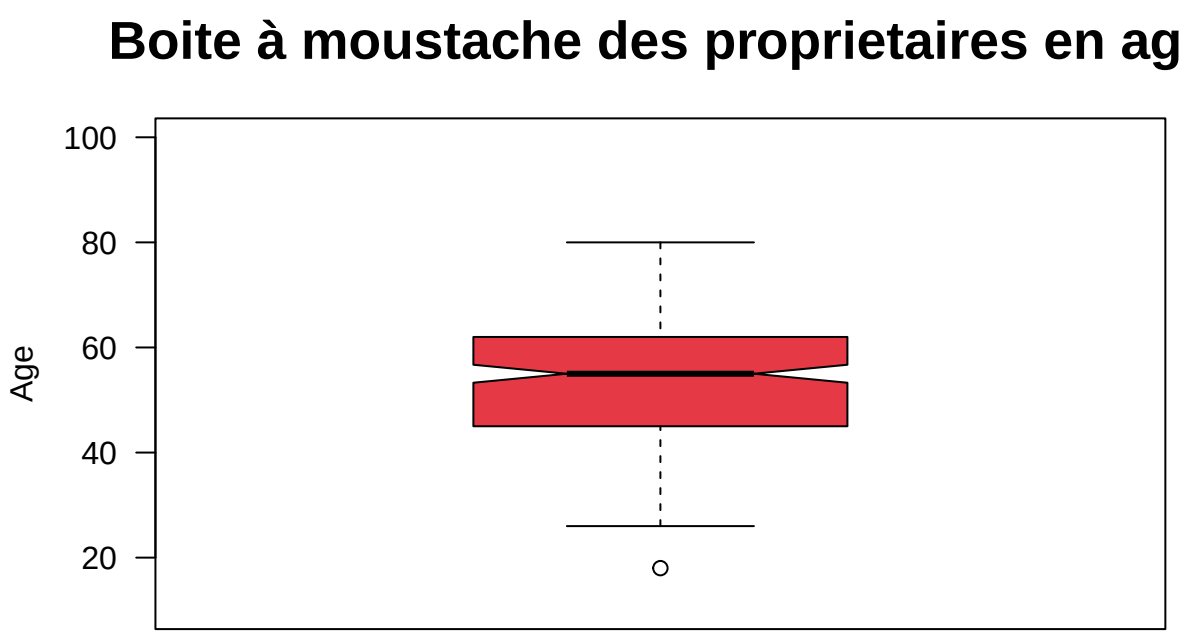
```
    panel.background = element_rect(fill = "#F0F0F0"),
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.position = "top"
  )
```




```
#
```

```
#-------------------------------------------------------------------------------
# Une boite a moustache avec les individus agés de moins 120 ans  pour rester dans une realité
# Du fait qu'il existe des valeurs exorbitante dans la variable age
#-------------------------------------------------------------------------------

projet$Age <- projet$q24

Boite_moustache =  projet %>%
filter( Age < 120 )

boxplot(Boite_moustache$Age,
```

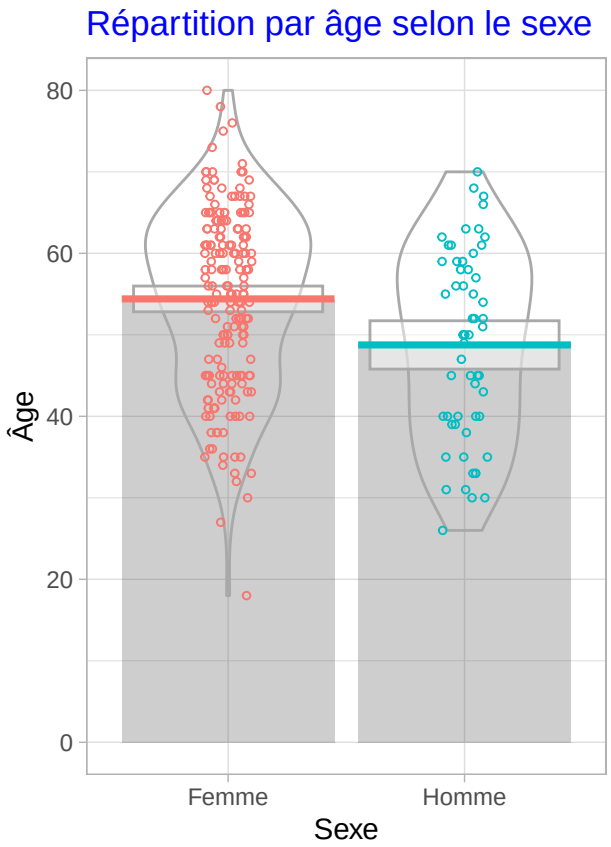
```
ylab = "Age",

main = "Boite à moustache des proprietaires en age",

col = "#e63946",

las = 1 ,

cex.main = 1.7,

sub = " Données : Base_Partie 1",
notch = TRUE,

ylim = c(10, 100)
)
```

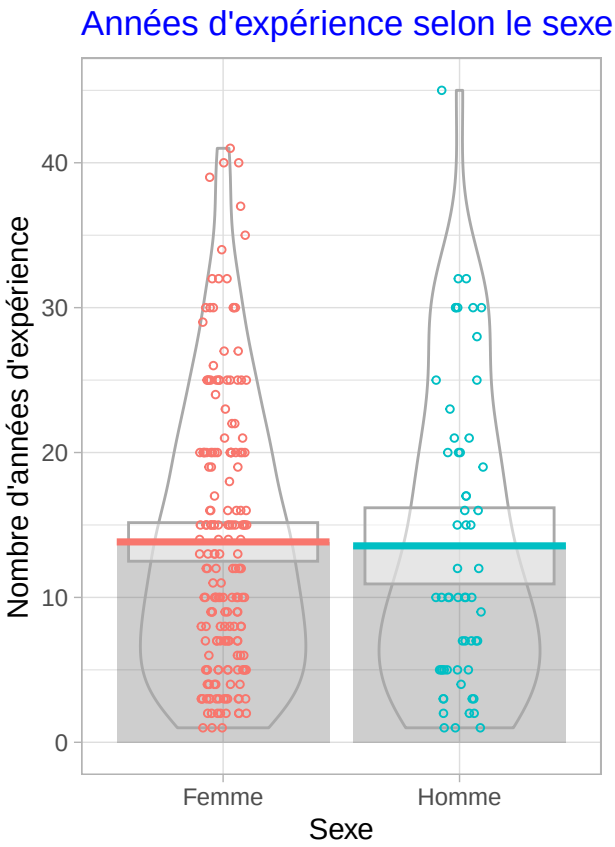


```r
#-------------------------------------------------------------------------------------------
```

```r
#-------------------------------------------------------------------------------------------
# Le package ggpirate qui permet de visualiser la distrubition d'une maniere plus clair
#-------------------------------------------------------------------------------------------

Repartition1 = projet %>%

  rename(age = q24) %>%

  filter( age < 120 )


Repartition2 = projet %>%

  rename(Annees_experience = q26) %>%

  filter( Annees_experience < 50 )


plot1 <- ggplot(Repartition1, aes(x = sexe, y = age)) +

  geom_pirate(aes(colour = sexe)) +

  xlab("Sexe") +

  ylab("Âge") +

  ggtitle("Répartition par âge selon le sexe") +

  theme_light()+

  theme(
    plot.title = element_text(color = "blue"))

plot2<- ggplot(Repartition2, aes(x = sexe, y = Annees_experience)) +
```

```r
  geom_pirate(aes(colour = sexe)) +

  xlab("Sexe") +

  ylab("Nombre d'années d'expérience") +

  ggtitle("Années d'expérience selon le sexe") +

  theme_light()+

  theme(
    plot.title = element_text(color = "blue"))

# ------------------------------------------------------------------------------
#Afficher les graphiques côte à côte sur une même ligne
#------------------------------------------------------------------------------

grid.arrange(plot1, plot2, ncol = 2)
```


Répartition par âge selon le sexe
80
60
40
20
0
Âge
Femme
Homme
Sexe


Années d'expérience selon le sexe
40
30
20
10
0
Nombre d'années d'expérience
Femme
Homme
Sexe

```
#-------------------------------------------------------------------------------
```

1.14 Transformer le data.frame en données géographiques dont l'objet sera nommé projet_map

```
#-------------------------------------------------------------------------------------------
# Importation de la base senegal qui contient les coordonnees geographiques au niveau 1 (les regions)
#-------------------------------------------------------------------------------------------

senegal <- st_read("SEN_adm1.shp", layer = "SEN_adm1", stringsAsFactors = FALSE)
```

```
## Reading layer 'SEN_adm1' from data source
##   'C:\Users\medeh\Documents\Projet_R\SEN_adm1.shp' using driver 'ESRI Shapefile'
## Simple feature collection with 14 features and 9 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -17.54319 ymin: 12.30786 xmax: -11.34247 ymax: 16.69207
## Geodetic CRS:  WGS 84
```

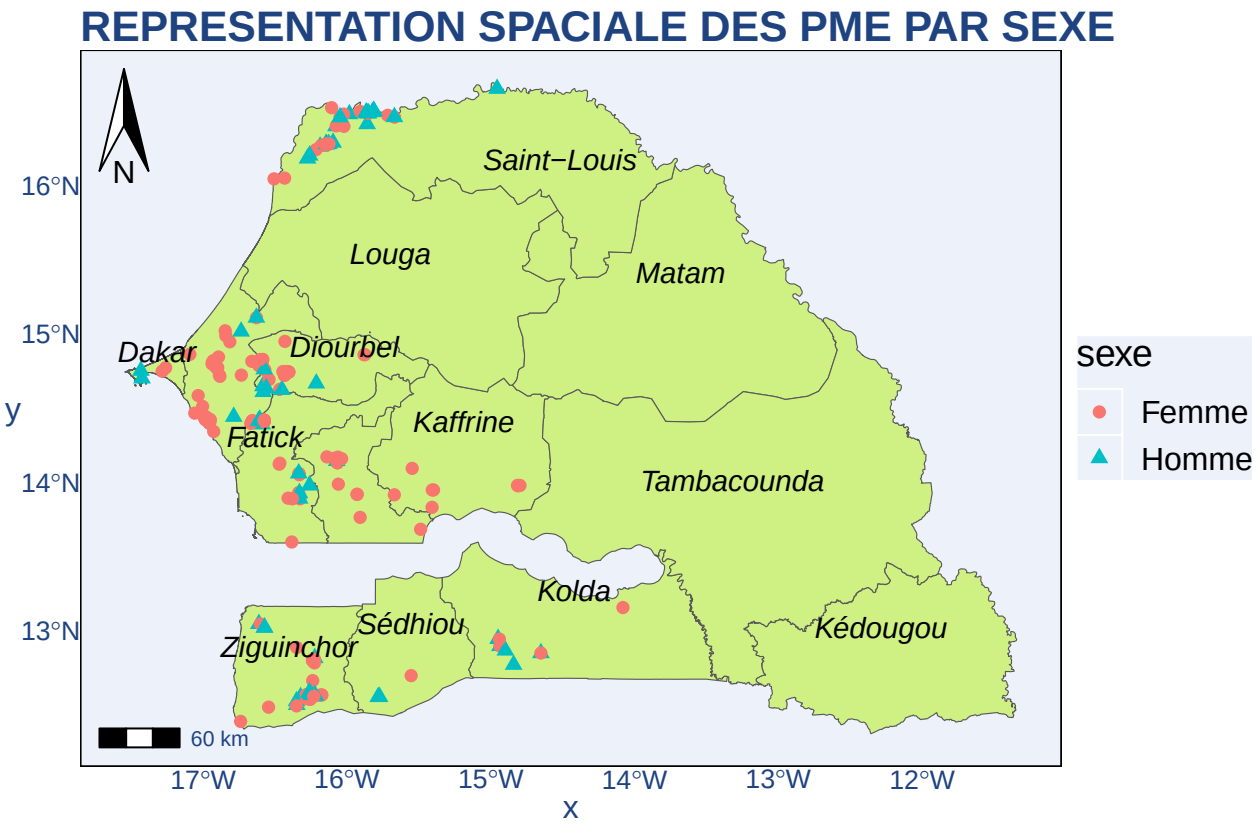
```
#-------------------------------------------------------------------------------------------
#   Creer la variable ID qui prend les regions et selection de ID et les coordonnees geometry
#-------------------------------------------------------------------------------------------

senegal$ID <- senegal$NAME_1
senegal <- senegal[,c(11,10)]



#-------------------------------------------------------------------------------------------
# Transformation de notre base en donnnes geometrique et specificer crs = st_crs(senegal)
#-------------------------------------------------------------------------------------------
projet_map <- st_as_sf(projet, coords = c("gps_menlongitude", "gps_menlatitude"),
                       crs = st_crs(senegal))


projet_map <- st_join(projet_map,senegal)
```

```r
#-------------------------------------------------------------------------------------
#             S'assurer qu'il s'agit d'un sf
#-------------------------------------------------------------------------------------
class(projet_map)
```

```
## [1] "sf"         "tbl_df"     "tbl"        "data.frame"
```

```r
#-------------------------------------------------------------------------------------
#  Representation de la carte avec geom_sf : on represente d'abord la carte du senegal
#      comme premiere couche avec la base sebegal
#        Ensuite les coorddones de nos PME avec quelques esthetique pour la beaute
#-------------------------------------------------------------------------------------

ggplot() +

  geom_sf(data = senegal, fill = "#CFF183") +

  geom_sf(data = projet_map, size = 2, aes(fill = sexe, col = sexe, shape = sexe)) +

  geom_sf_text(data = senegal, aes(label = ID), vjust = -0.5,
               check_overlap = TRUE, fontface = "italic", color = "black") +

  labs(title = "REPRESENTATION SPACIALE DES PME PAR SEXE")+

  theme_void()+

   theme(
    panel.background = element_rect(fill = "#ECF1FA"),
      # Ajouter une couleur de fond

    plot.title = element_text(color = "#1E4380", size = 16, face = "bold"),
    legend.position = "right",

    legend.title = element_text(size = 14),

    legend.text = element_text(size = 12),
```

```
  legend.background = element_rect(fill = "#ECF1FA", color = "white"),

  legend.key = element_rect(color = "white"),

  axis.text = element_text(color = "#1E4380", size = 10),

  axis.title = element_text(color = "#1E4380", size = 12)
)+

annotation_scale( location = "bl",width_hint = 0.1,

text_col = "#1E4380",text_size = 1, bar_col = "back")+

  annotation_north_arrow(location = "tl", which_north = "true",
                         heigt = unit(0.05,"npc"),width = unit(0.05,"npc"))
```

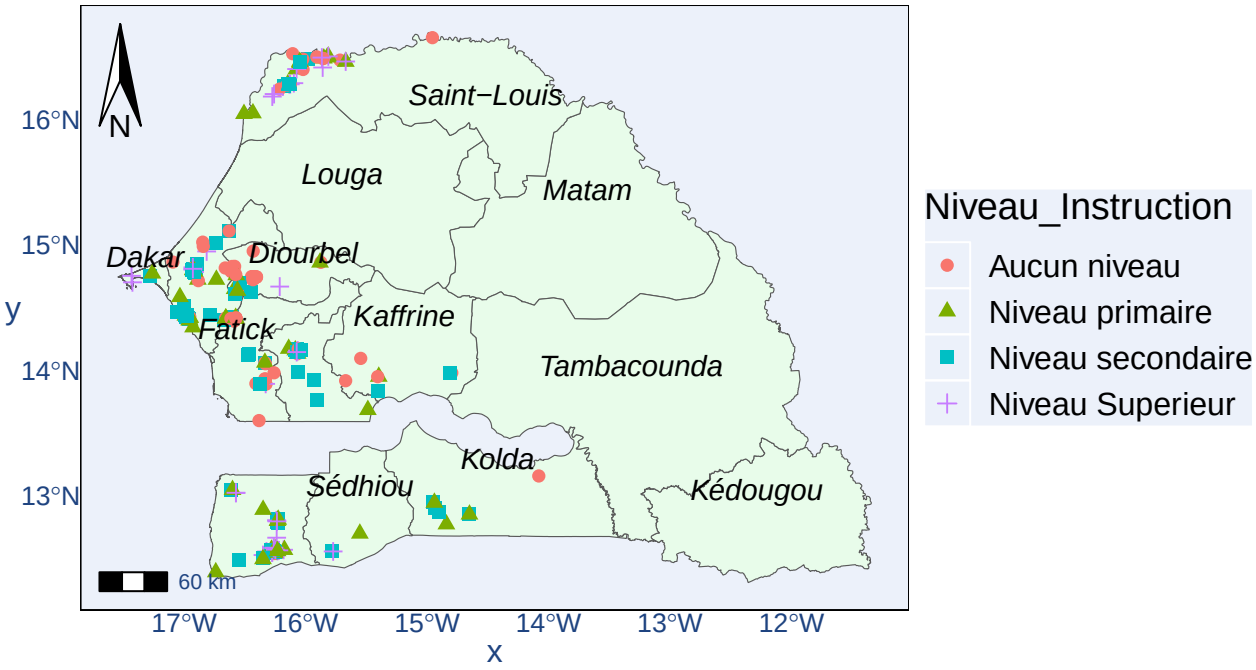


1.15 La réprésentation spatiale des PME suivant le niveau d'instruction

```
#-------------------------------------------------------------------------------------------
#   Le même jeu que le precedent
#-------------------------------------------------------------------------------------------

projet_map = projet_map %>% rename(Niveau_Instruction = q25)

ggplot() +

  geom_sf(data = senegal, fill = "#E8FCEA") +

  geom_sf(data = projet_map, size = 2, aes(fill =
                                             Niveau_Instruction, col = Niveau_Instruction, shape =
                                             Niveau_Instruction)) +

  geom_sf_text(data = senegal, aes(label = ID),

               vjust = -0.5, check_overlap = TRUE, fontface =

                 "italic", color = "black") +

  labs(title = "REPRESENTATION PAR NIVEAU D'INDTRUCTION")+

  theme_void()+

   theme(

    panel.background = element_rect(fill = "#ECF1FA"),

    # Ajouter une couleur de fond

    plot.title = element_text(color = "#1E4380", size = 16, face = "bold"),
```

```
  legend.position = "right",

  legend.title = element_text(size = 14),

  legend.text = element_text(size = 12),

  legend.background = element_rect(fill = "#ECF1FA", color = "white"),

  legend.key = element_rect(color = "white"),

  axis.text = element_text(color = "#1E4380", size = 10),

  axis.title = element_text(color = "#1E4380", size = 12)
)+

annotation_scale( location = "bl",width_hint = 0.1,

text_col = "#1E4380",text_size = 1, bar_col = "back")+

  annotation_north_arrow(location = "tl", which_north =
                           "true",

                       heigt = unit(0.05,"npc"),width = unit(0.05,"npc"))
```

REPRESENTATION PAR NIVEAU D'INDTRUCTION



1.16 Faites une analyse spatiale de votre choix

```
#-----------------------------------------------------------------------------------------
#  Regouper par filiere avec uniquement ceux qui ceux qui sont specialisés dans
#   les filieres : On a dont 4 tableaux Tab1...Tab2
#-----------------------------------------------------------------------------------------

Tab1 = projet %>%
filter(filiere_1 == 1) %>%
dplyr::mutate(FILIERE = ifelse(filiere_1 == 1, "Arachide"))



Tab2 = projet %>%
  filter(filiere_2 == 1) %>%
  dplyr::mutate(Anacarde = ifelse(filiere_2 == 1, "Anacarde"))



Tab3 = projet %>%
  filter(filiere_3 == 1) %>%
  dplyr::mutate(Mangue = ifelse(filiere_3 == 1, "Mangue"))



Tab4 = projet %>%filter(filiere_4 == 1) %>%
  dplyr::mutate(Riz = ifelse(filiere_4 == 1, "Riz"))



#-----------------------------------------------------------------------------------------
# Importation de la base senegal qui contient les coordonnees geographiques
#    au niveau 2 (les departements)
#   Et le même jeu comme les deux carte precedentes
#-----------------------------------------------------------------------------------------

sene <- st_read("SEN_adm2.shp", layer = "SEN_adm2", stringsAsFactors = FALSE)
```

```
## Reading layer ‘SEN_adm2’ from data source
##   ‘C:\Users\medeh\Documents\Projet_R\SEN_adm2.shp’ using driver ‘ESRI Shapefile’
## Simple feature collection with 45 features and 11 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -17.54319 ymin: 12.30786 xmax: -11.34247 ymax: 16.69207
## Geodetic CRS:  WGS 84
```

```
Tab1 <- st_as_sf(Tab1, coords = c("gps_menlongitude", "gps_menlatitude"), crs = st_crs(sene))
```

```
Tab2 <- st_as_sf(Tab2, coords = c("gps_menlongitude", "gps_menlatitude"), crs = st_crs(sene))
```

```
Tab3 <- st_as_sf(Tab3, coords = c("gps_menlongitude", "gps_menlatitude"), crs = st_crs(sene))
```

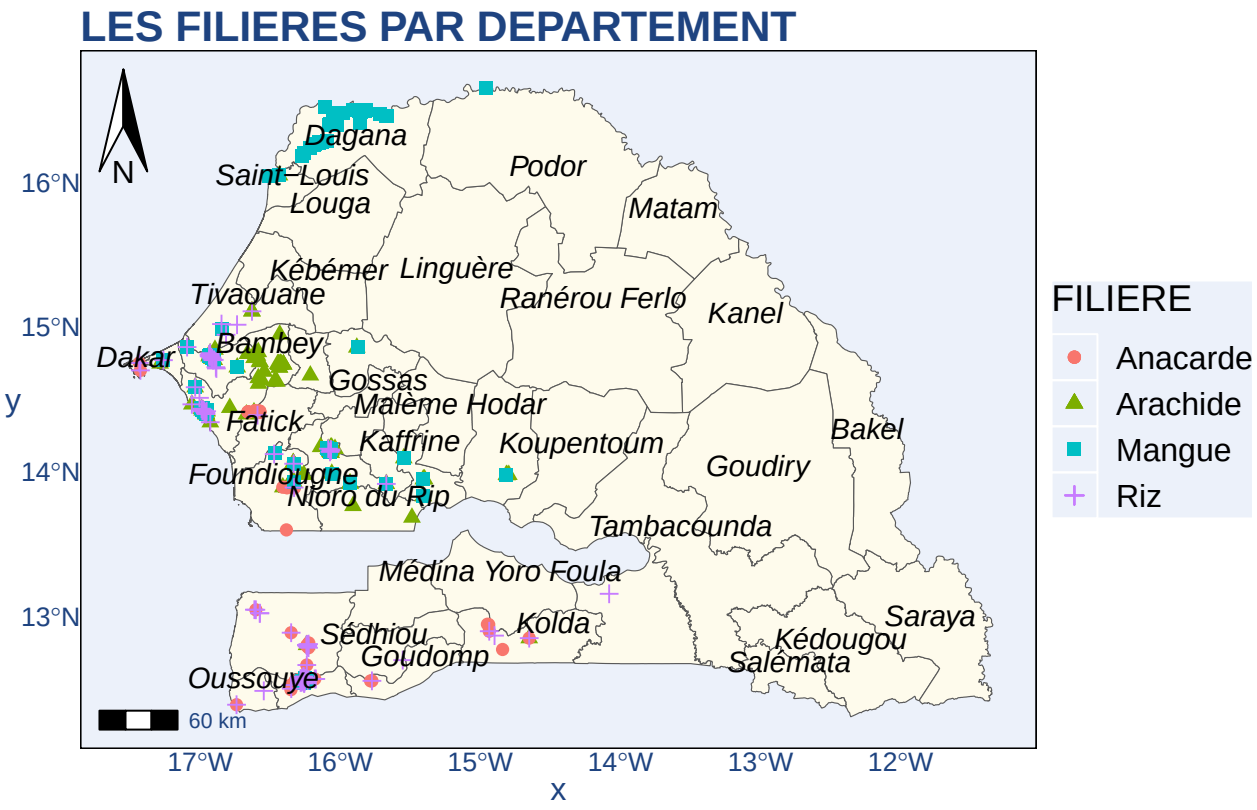
```
Tab4 <- st_as_sf(Tab4, coords = c("gps_menlongitude", "gps_menlatitude"), crs = st_crs(sene))
```

```
Tab1 <- st_join(Tab1,sene)
Tab2 <- st_join(Tab2,sene)
Tab3 <- st_join(Tab3,sene)
Tab4 <- st_join(Tab4,sene)
```

```
ggplot() +

  geom_sf(data = sene, fill = "#FEFBEC") +

  geom_sf(data = Tab1, size = 2, aes(fill = FILIERE, col = FILIERE , shape = FILIERE  ))+
```

```r
geom_sf(data = Tab2, size = 2, aes(fill = Anacarde,col = Anacarde,shape = Anacarde))+

geom_sf(data = Tab3, size = 2, aes(fill = Mangue,col = Mangue,shape = Mangue))+

geom_sf(data = Tab4, size = 2, aes(fill = Riz,col = Riz, shape = Riz))+

geom_sf_text(data = sene, aes(label = NAME_2),
             vjust = -0.1, check_overlap = TRUE, fontface = "italic", color = "black") +

labs(title = "LES FILIERES PAR DEPARTEMENT")+

theme_void()+

 theme(
  panel.background = element_rect(fill = "#ECF1FA"),  # Ajouter une couleur de fond
  plot.title = element_text(color = "#1E4380", size = 16, face = "bold"),

  legend.position = "right",

  legend.title = element_text(size = 14),

  legend.text = element_text(size = 12),

  legend.background = element_rect(fill = "#ECF1FA", color = "white"),

  legend.key = element_rect(color = "white"),

  axis.text = element_text(color = "#1E4380", size = 10),

  axis.title = element_text(color = "#1E4380", size = 12)
)+

annotation_scale( location = "bl",width_hint = 0.1,

text_col = "#1E4380",text_size = 1, bar_col = "back")+
```

```
    annotation_north_arrow(location = "tl", which_north = "true",
                           heigt = unit(0.05,"npc"),width = unit(0.05,"npc"))
```



PARTIE II

2 Nettoyage et gestion des données

2.1 Nomination de la variable country_destination en destination et définition des valeurs négatives comme manquantes

```
#-------------------------------------------------------------------------------------
# lecture de la feuille 1 du fichier Excel "Base_Partie 2.xlsx" :IMPORTATION
# Utilisation de la fonction rename pour renommer
#-------------------------------------------------------------------------------------

Feuille1 <- read_excel("Base_Partie 2.xlsx")

Feuille1 <- Feuille1 %>%

rename(destination = country_destination) %>%

mutate(across(everything(), ~ ifelse(. < 0, NA, .)))

# ou encore Feuille1[Feuille1 < 0] <- NA

Feuille1 %>%

  head()
```

```
## # A tibble: 6 x 10
##      id starttime endtime enumerator district   age   sex children_num intention
##   <dbl>     <dbl>   <dbl>      <dbl>    <dbl> <dbl> <dbl>        <dbl>     <dbl>
## 1     2    1.55e9  1.55e9          6        1    33     1            1         1
```

```
## 2     3    1.55e9  1.55e9          6        1    43     0            5         1
## 3     4    1.55e9  1.55e9          6        1    28     0            0         1
## 4     7    1.55e9  1.55e9          8        3    24     0            0         1
## 5     8    1.55e9  1.55e9          8        3    29     0            0         1
## 6    10    1.55e9  1.55e9          8        6    22     1            0         1
## # i 1 more variable: destination <dbl>
```

```
#-------------------------------------------------------------------------------
```

2.2 Créer une nouvelle variable contenant des tranches d'âge de 5 ans en utilisant la variable "age"

```
#-------------------------------------------------------------------------------
#  1)  Attacher la base
#    Definir un vecteur textlab qui prends les chaines de charactere "[a,b["
#    Creer la nouvelle variables de groupe d'age avec cut

attach(Feuille1)

min(age)
```

```
## [1] 15
```

```
textlab <-c()

largeur <- 5

nbcat <- 9

for( i in 3 : (nbcat - 1 )){

textlab[i+1] <- paste("[",as.character(i*largeur),",",as.character((i+1)*largeur),"[",sep="")}

textlab <- textlab[complete.cases(textlab)]
```

```
# Supprimer les valeur manquantes pour une dimension identique



Feuille1 = Feuille1 %>%

  mutate( group_age = cut ( age, breaks = seq(from = 15, to = nbcat*largeur,
                                              by = largeur), labels = textlab),.after = age)


Feuille1 %>%

  head()
```

```
## # A tibble: 6 x 11
##      id starttime endtime enumerator district   age group_age   sex children_num
##   <dbl>     <dbl>   <dbl>      <dbl>    <dbl> <dbl> <fct>     <dbl>        <dbl>
## 1     2    1.55e9  1.55e9          6        1    33 [30,35[       1            1
## 2     3    1.55e9  1.55e9          6        1    43 [40,45[       0            5
## 3     4    1.55e9  1.55e9          6        1    28 [25,30[       0            0
## 4     7    1.55e9  1.55e9          8        3    24 [20,25[       0            0
## 5     8    1.55e9  1.55e9          8        3    29 [25,30[       0            0
## 6    10    1.55e9  1.55e9          8        6    22 [20,25[       1            0
## # i 2 more variables: intention <dbl>, destination <dbl>
```

2.3 Créer une nouvelle variable contenant le nombre d'entretiens réalisés par chaque agent recenseur

```
#-----------------------------------------------------------------------------------------------
#   La fonction count pour compter le nombre de repetition de
#   la variable enumerator et l'ajouter a la variable nombre_entretien
#-----------------------------------------------------------------------------------------------

Entretien = Feuille1 %>%
```

```
count(enumerator, sort = TRUE , name = "nombre_entretien")


Entretien
```

```
## # A tibble: 16 x 2
##    enumerator nombre_entretien
##         <dbl>            <int>
##  1          4                9
##  2         20                9
##  3         13                8
##  4          7                7
##  5         11                7
##  6          5                6
##  7          8                6
##  8          9                6
##  9         14                6
## 10         17                6
## 11         18                6
## 12          1                5
## 13          6                5
## 14         10                5
## 15         12                5
## 16         15                1
```

2.4 Créer une nouvelle variable qui affecte aléatoirement chaque répondant à un groupe de traitement (1) ou de controle (0)

```
#-------------------------------------------------------------------------------------
#    set.seed pour fixer l'alea
#-------------------------------------------------------------------------------------

set.seed(012)
```

```r
Feuille1 = Feuille1 %>%

mutate( traitement = sample(c(0,1), size = nrow(Feuille1), replace = TRUE))



Feuille1 %>%

  head()
```

```
## # A tibble: 6 x 12
##      id starttime endtime enumerator district   age group_age   sex children_num
##   <dbl>     <dbl>   <dbl>      <dbl>    <dbl> <dbl> <fct>     <dbl>        <dbl>
## 1     2    1.55e9  1.55e9          6        1    33 [30,35[       1            1
## 2     3    1.55e9  1.55e9          6        1    43 [40,45[       0            5
## 3     4    1.55e9  1.55e9          6        1    28 [25,30[       0            0
## 4     7    1.55e9  1.55e9          8        3    24 [20,25[       0            0
## 5     8    1.55e9  1.55e9          8        3    29 [25,30[       0            0
## 6    10    1.55e9  1.55e9          8        6    22 [20,25[       1            0
## # i 3 more variables: intention <dbl>, destination <dbl>, traitement <dbl>
```

2.5 Fusion (feuille 1) & (feuille 2)

```r
#-------------------------------------------------------------------------------------------
# lecture de la feuille 2 du fichier Excel "Base_Partie 2.xlsx" :IMPORTATION
#-------------------------------------------------------------------------------------------

Feuille2 <- read_excel("Base_Partie 2.xlsx",
sheet = "district")

#-------------------------------------------------------------------------------------------
#             Fusion avec left_join
#-------------------------------------------------------------------------------------------

Fusion_feuille = left_join(Feuille1,Feuille2, by = "district")
```

```r
Fusion_feuille %>%

  head()
```

```
## # A tibble: 6 x 13
##      id starttime endtime enumerator district   age group_age   sex children_num
##   <dbl>     <dbl>   <dbl>      <dbl>    <dbl> <dbl> <fct>     <dbl>        <dbl>
## 1     2    1.55e9  1.55e9          6        1    33 [30,35[       1            1
## 2     3    1.55e9  1.55e9          6        1    43 [40,45[       0            5
## 3     4    1.55e9  1.55e9          6        1    28 [25,30[       0            0
## 4     7    1.55e9  1.55e9          8        3    24 [20,25[       0            0
## 5     8    1.55e9  1.55e9          8        3    29 [25,30[       0            0
## 6    10    1.55e9  1.55e9          8        6    22 [20,25[       1            0
## # i 4 more variables: intention <dbl>, destination <dbl>, traitement <dbl>,
## #   population <dbl>
```

2.6 Calculer la durée de l'entretien et indiquer la durée moyenne de l'entretien par enquêteur

```r
attach(Fusion_feuille)

#-----------------------------------------------------------------------------------------
# Convertir les colonnes "starttime" et "endtime" en objets de type POSIXct
# pour pouvoir effectuer des calculs de durée :
#-----------------------------------------------------------------------------------------
starttime <- as.POSIXct(starttime)

endtime <- as.POSIXct(endtime)

Fusion_feuille %>% names()
```

```
##  [1] "id"           "starttime"    "endtime"      "enumerator"   "district"
##  [6] "age"          "group_age"    "sex"          "children_num" "intention"
```

```
## [11] "destination"  "traitement"   "population"
```

```
# Calculer la durée de l'entretien en soustrayant la colonne "starttime" de la colonne "endtime"
Fusion_feuille %>%
  mutate(duree =difftime(endtime, starttime, units = "mins")) %>%
  group_by(enumerator) %>%

  summarise(duree_moyenne = mean(duree, na.rm = TRUE))
```

```
## # A tibble: 16 x 2
##    enumerator duree_moyenne
##         <dbl> <drtn>
##  1          1  68.14667 mins
##  2          4  36.48333 mins
##  3          5  33.55833 mins
##  4          6  25.84667 mins
##  5          7  37.16429 mins
##  6          8  40.13056 mins
##  7          9 114.76667 mins
##  8         10  55.27667 mins
##  9         11  33.48333 mins
## 10         12  48.16667 mins
## 11         13  31.59583 mins
## 12         14  25.56111 mins
## 13         15  28.65000 mins
## 14         17  29.28611 mins
## 15         18  36.85833 mins
## 16         20  28.76852 mins
```

2.7 Les variables de l'ensemble de données en ajoutant le préfixe "endline_"

```
#-------------------------------------------------------------------------------
# Définissez le préfixe pour les nouvelles colonnes
#-------------------------------------------------------------------------------
```

```r
prefixe <- "endline_"

#-------------------------------------------------------------------------------------------
# Renommez les colonnes en ajoutant le préfixe
#-------------------------------------------------------------------------------------------

noms_nouveaux <- lapply(names(Fusion_feuille), function(col)

  paste0(prefixe, col))

names(Fusion_feuille) <- unlist(noms_nouveaux)



Fusion_feuille %>%
names()
```

```
##  [1] "endline_id"           "endline_starttime"    "endline_endtime"
##  [4] "endline_enumerator"   "endline_district"     "endline_age"
##  [7] "endline_group_age"    "endline_sex"          "endline_children_num"
## [10] "endline_intention"    "endline_destination"  "endline_traitement"
## [13] "endline_population"
```

2.8 Analyse et visualisation des données

2.9 Créez un tableau récapitulatif contenant l'âge moyen et le nombre moyen d'enfants par district

```r
#-------------------------------------------------------------------------------------------
#   Meme principe comme au niveau de l'age
#-------------------------------------------------------------------------------------------

tableau_recap <- Fusion_feuille %>%
```

```
  group_by(endline_district) %>%

summarise(age_moyen = mean(endline_age, na.rm = TRUE),

          nbre_moyen_enfant = mean(endline_children_num, na.rm = TRUE)) %>%

  mutate(age_moyen = round(age_moyen),

         nbre_moyen_enfant = round(nbre_moyen_enfant))

tableau_recap
```

```
## # A tibble: 8 x 3
##   endline_district age_moyen nbre_moyen_enfant
##              <dbl>     <dbl>             <dbl>
## 1                1        30                 2
## 2                2        63                 1
## 3                3        26                 0
## 4                4        26                 0
## 5                5        24                 0
## 6                6        23                 0
## 7                7        28                 0
## 8                8        25                 1
```

```
#-------------------------------------------------------------------------------
```

\textcolor{blue}{\subsection{Testez si la différence d'âge entre les sexes est statistiquement significative au niveau de 5 %}}

```
#-------------------------------------------------------------------------------
#  Renommos les observations de la variable sexe
#-------------------------------------------------------------------------------

Fusion_feuille_test <- Fusion_feuille %>%

dplyr::mutate(endline_sex = ifelse(endline_sex == "0", "homme", "femme"))
```

```r
#-------------------------------------------------------------------------------
# la fonction tbl_summary comme dans la premiere partie avec un filtre de l'age #999
#-------------------------------------------------------------------------------

Fusion_feuille_test %>%

filter(endline_age != 999) %>%
tbl_summary(
include = c("endline_age","endline_sex"),
by = endline_sex,
statistic = endline_age ~ " {mean} [{sd}]",
label = endline_age ~ "AGE",
digits = list(endline_age ~ 0)
) %>%

add_difference() %>%

add_overall(
last = TRUE
) %>%

bold_labels() %>%

italicize_levels() %>%

modify_header(update = list( label ~ "**Variable**",
all_stat_cols(stat_0 = FALSE) ~ "_{level}_ (n={n}, {style_percent(p)}%)",
stat_0 ~ "**TOTAL** (n={N})",
p.value ~ "**Test de comparaison** (p-valeur)"
)) %>%

  bold_labels() %>%

italicize_levels() %>%
```

```
modify_spanning_header(
all_stat_cols(stat_0 = FALSE) ~ "**SEXE**"
) %>%
as_flex_table() %>%

  fontsize(size = 8) %>%

  width(width = 1)
```

	SEXE					
Variable	*femme* (n=10, 10%)[1]	*homme* (n=86, 90%)[1]	**Difference**[2]	**95% IC**[23]	**Test de comparaison** (p-valeur)[2]	**TOTAL** (n=96)[1]
AGE	22 [5]	26 [6]	-4,0	-7,9 – -0,05	0,047	26 [6]

[1] Moyenne [ET]

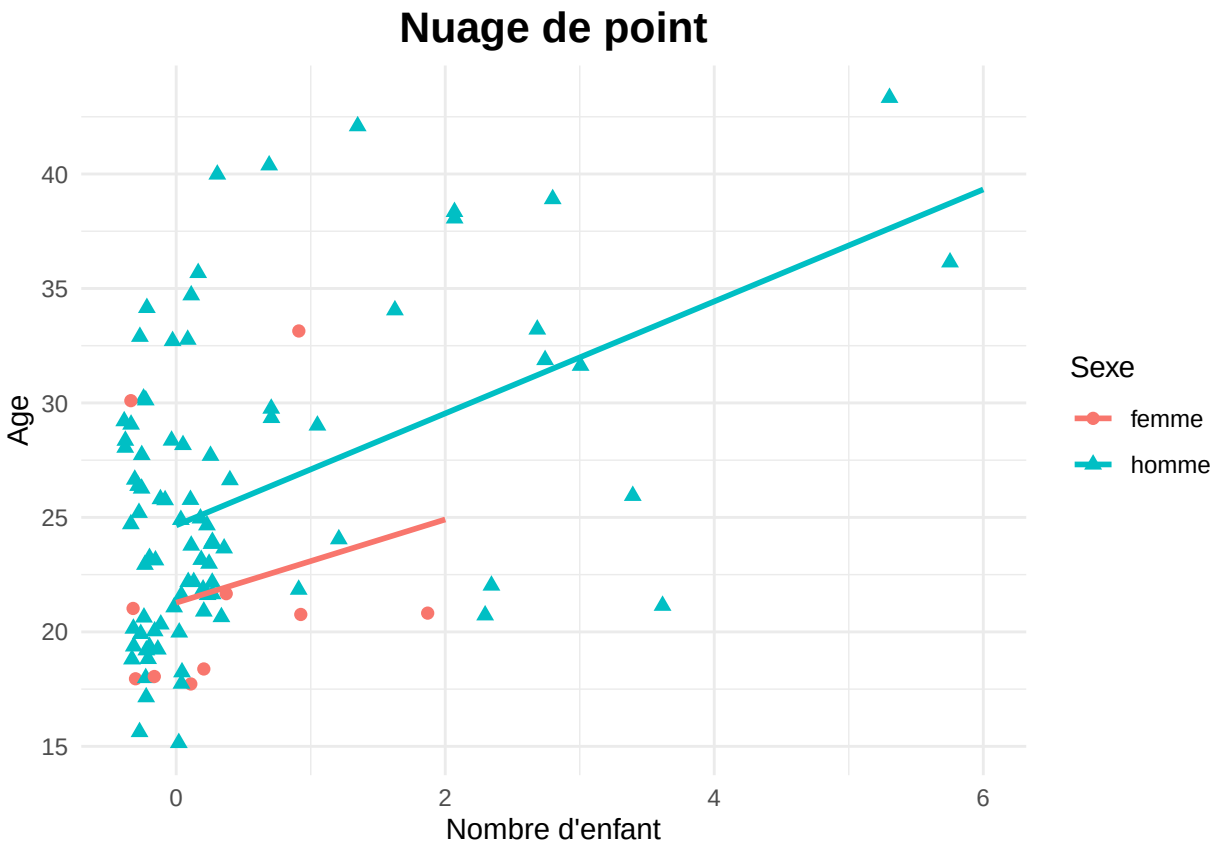
[2]test de Student

[3]IC = intervalle de confiance

2.10 Création de nuage de points de l'âge en fonction du nombre d'enfants

```
#-------------------------------------------------------------------------------
# Nuage de point et droit de regression par sexe
#-------------------------------------------------------------------------------


Nuage = Fusion_feuille_test %>%
```

```r
  filter(endline_age != 999)

ggplot(Nuage,aes( x = endline_children_num , y = endline_age,col= endline_sex )) +

geom_jitter(aes(shape = endline_sex),size = 2)+

  geom_smooth(method = "lm", se = FALSE)+

labs(x="Nombre d'enfant", y = "Age", title = "Nuage de point")+

labs( col = "Sexe", shape = "Sexe") +

theme_minimal()+

theme(plot.title = element_text(color = "black", size = 16, face = "bold", hjust = 0.5 ))
```



2.11 l'effet de l'appartenance au groupe de traitement sur l'intention de migrer

```
modele <- stats::lm(endline_intention ~ endline_traitement, Fusion_feuille)



modele %>%
  gtsummary::tbl_regression(
    label = list(endline_traitement ~ "TRAITEMENT")
  )%>%
  as_flex_table()
```

Caractéristique	Beta	95% IC[1]	p-valeur
TRAITEMENT	0,33	-0,37 – 1,0	0,3

[1]IC = intervalle de confiance

2.12 Tableau de régression avec 3 modèles

```
Modele_A = modele %>%
  gtsummary::tbl_regression()

Modele_B = stats::lm(data =Fusion_feuille,  endline_traitement ~ endline_age + endline_sex) %>%
   gtsummary::tbl_regression()

Modele_C = stats::lm(data =Fusion_feuille,  endline_traitement ~ endline_age + endline_sex+ endline_dis
  gtsummary::tbl_regression()

gtsummary::tbl_stack(
  list(Modele_A,
```

```
      Modele_B,
      Modele_C),
  group_header = c("MODELE A", "MODELE B", "MODELE C")
) %>%
  as_flex_table()
```

Group	Caractéristique	Beta	95% IC[1]	p-valeur
MODELE A	endline_traitement	0,33	-0,37 – 1,0	0,3
MODELE B	endline_age	0,00	0,00 – 0,00	0,5
	endline_sex	0,15	-0,18 – 0,48	0,4
MODELE C	endline_age	0,00	0,00 – 0,00	0,5
	endline_sex	0,14	-0,19 – 0,47	0,4
	endline_district	-0,02	-0,06 – 0,02	0,4

[1]IC = intervalle de confiance