TP Sirene

Benoît Blanc

03/01/2020

Contents

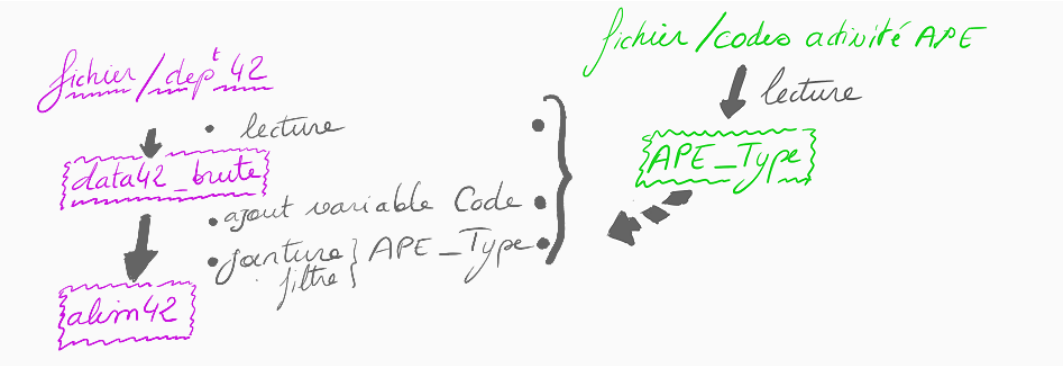
0.1 R Markdown

Ce fichier montre les résultats des questions du TP disponible ici. J'ai choisi de ne pas afficher tout le temps le code R que j'ai écrit pour répondre aux questions. Les résultats sont tous affichés dans le document. Le code R Markdown est disponible sur mon compte GitHub. J'ai utilisé les données de Janvier 2020 pour les entreprises.

Ce TP consiste à travailler sur des données de la base Sirene de l'INSEE, mise à disposition sur data.gouv.fr, et qui répertorie l'ensemble des entreprises et établissements actifs en France. Les métadonnées associées à cette base sont en partie décrites dans ce tableau.

---

1 Mise en place



Pour réaliser ce TP, nous avons besoin de toutes les bibliothèques suivantes :

```
library(dplyr)
library(readr)
library(stringr)
library(purrr)
library(lubridate)
library(scales)
library(tidyr)
library(ggplot2)
library(sf)
library(plotly)
library(prettydoc)
library(rmdformats)
```

1.1 Installations, téléchargements, premiers tests sur le département 42

- Téléchargez les données du département 42, geo_siret_42.csv dans ce répertoire pour le mois de Janvier 2020 et dézippez le dossier sur votre machine.
- Téléchargez la table qui renseigne les codes correspondant à l'Activité Principale de l'Etablissement (APE)APE_Type.csv

1.2 Lecture de tableaux de données

- Depuis RStudio, créez un projet qui comprendra l'ensemble des données et documents nécessaires à réaliser l'ensemble des traitements qui vous seront demandés pour ce TP.
- Créez le data.frame data42 en lisant la table geo-siret_42.csv.

```
data42 <- read_csv("data/geo_siret_42.csv")
```

- Créez l'objet APE_Type en lisant le fichier relatif aux codes d'APE.

```
APE_Type <- read_csv("data/APE_Type.csv")
```

1.3 Code et types d'activités => commerces alimentaires

- Combien d'entreprises ont un nom (enseigne1Etablissement) qui comprend le terme "BOULANGERIE"?

Il y a 84 entreprises qui comprennent le terme "BOULANGERIE" dans leur nom.

- Ajoutez une variable Code à votre table en ne conservant que les quatre premiers caractères de la variable activitePrincipaleEtablissement (cela correspond à un pattern "^...", à savoir le début de chaîne de caractère suivi de quatre caractères quelconques -cf ce billet de blog sur les expressions régulières-).

```
data42 <- mutate(data42, Code = as.numeric(str_extract(activitePrincipaleEtablissement,
    "^....")))
```

- Filtrez les lignes de data42 pour ne retenir que celles pour lesquelles l'APE correspond aux commerces "alimentaires" -alimentation, boisson, restaurant, bar- (voir la liste contenue dans le fichier APE_Type).
- Stockez le résultat de ces opérations dans un objet alim42.
- Réalisez une jointure entre data42_alim (variable codeAPE) et APE_Type (variable Code), de manière à compléter alim42 avec les types de commerces (variables Type et TypeAbreg).

```
alim42 <- left_join(alim42, APE_Type, by = "Code")
```

1.4 Résumé, classement

- Quelles sont les 3 communes de votre base de données qui comptent le plus de magasins alimentaires?

Les trois communes comptant le plus de commerces alimentaires sont : SAINT-ETIENNE, ROANNE, SAINT-CHAMOND

- Pour les communes qui ne comptent qu'un seul commerce "alimentaire", de quel type est-il, le plus fréquemment?

Les communes ne comptant qu'un seul commerce "alimentaire" sont au nombre de 17. Avec en détail : BOYER, CEZAY, LA CHAPELLE-EN-LAFAYE, DEBATS RIVIERE D'ORPRA, GREZIEUX-LE-FROMENTAL, JARNOSSE, LURE, PINAY, ROCHE, SAINTE-AGATHE-EN-DONZY, SAINT-LAURENT-ROCHEFORT, SAINT-PIERRE-LA-NOAILLE, SAINT-PRIEST-LA-VETRE, SAINT-SIXTE, SAINT-THOMAS-LA-GARDE, SOUTERNON, URBISE

Lorsqu'il y a un seul commerce alimentaire dans une commune, celui-ci est de type Restaurants et services de restauration mobile.

- Quelles communes de plus de 100 commerces comptent au moins 10 commerces de type "viande"?

```
morethan100commerces <- group_by(data42, codeCommuneEtablissement,
    libelleCommuneEtablissement) %>% summarise(nb_commerces = n())
morethan100commerces <- filter(morethan100commerces, nb_commerces >=
    100)
```

Les communes de plus de 100 commerces sont ANDREZIEUX-BOUTHEON, LE CHAMBON-FEUGEROLLES, CHARLIEU, CHAZELLES-SUR-LYON, LE COTEAU, FEURS, FIRMINY, MONTBRISON, LA RICAMARIE, RIVE-DE-GIER, ROANNE, ROCHE-LA-MOLIERE, SAINT-CHAMOND, SAINT-ETIENNE, SAINT-PRIEST-EN-JAREZ, SAINT-JUST-SAINT-RAMBERT, LA TALAUDIERE

```
morethan10viandes <- filter(data42, TypeAbreg == "viande") %>%
    group_by(codeCommuneEtablissement, libelleCommuneEtablissement) %>%
    summarise(nb_commerces_viande = n())
morethan10viandes <- filter(morethan10viandes, nb_commerces_viande >=
    10)
```

Les communes qui ont plus de 10 commerces de type 'viande' sont FEURS, MONTBRISON, ROANNE, SAINT-ETIENNE, LA TALAUDIERE

On retrouve les 5 communes ayant plus de 10 commerces de type 'viande' dans les villes ayant plus de 100 commerces. On a donc la réponse. Mais pour s'assurer, on peut réaliser une jointure entre les 2 tableaux :

```
more100commerces10viandes <- left_join(morethan10viandes, morethan100commerces,
    by = c("codeCommuneEtablissement", "libelleCommuneEtablissement"))
```

On retrouve le même résultat : FEURS, MONTBRISON, ROANNE, SAINT-ETIENNE, LA TALAUDIERE

---

2 Rapport, statistiques descriptives

A ce stade, votre script commence à être un peu long et (peut-être) un peu désordonné… Ne serait-ce pas plus agréable de continuer votre projet sous la forme d'un rapport Rmarkdown? (Ne répondez pas à cette question, elle est rhétorique…).

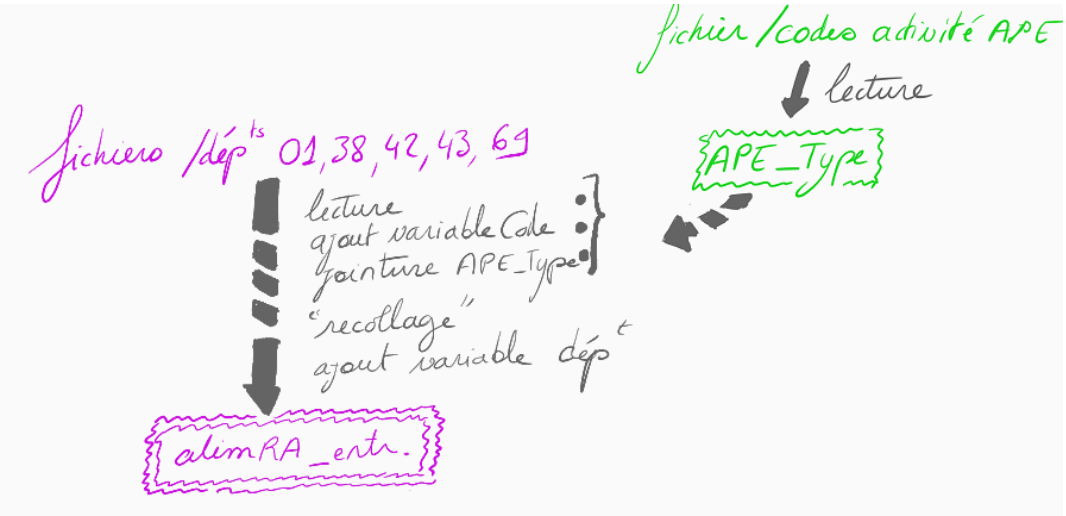
Créez un document _____.Rmd, structurez-le avec quelques titres, et répartissez les différents morceaux de code de votre script de manière pertinente dans différents chunks. Vous pouvez maintenant rédiger des paragraphes en y intégrant des éléments de réponses aux questions posées précédemment. Rédigez un petit paragraphe pour nommer les 3 communes qui comptent le plus d'entreprises (exercice précédent) en utilisant l'insertion d'"inline chunks".

Les 3 communes qui comptent le plus d'entreprises sont SAINT-ETIENNE, ROANNE, SAINT-CHAMOND.

A partir de maintenant, votre document de travail sera un document'_____.Rmd' et non le script que vous avez créé initialement…

---

3 Programmation: automatisation pour plusieurs départements



3.1 Fonction

Pour obtenir la table alim42, vous avez réalisé un certain nombre d'opérations. On voudrait réaliser l'ensemble de ces opérations pour les 5 départements suivants:

- l'Ain (01)
- l'Isère (38)
- la Loire (42)
- la Haute-Loire (43)
- le Rhône (69)

Réutilisez les commandes que vous avez mises au point sur data42 pour écrire une fonction get_clean_data() qui réalisera l'ensemble de ces opérations sur le département de votre choix. L'input correspondra à un numéro de département (c'est-à-dire que vous pourrez utiliser la fonction en faisant, par exemple get_clean_data("01")).

Pour lire le fichier, il faudra indiquer son chemin… Pensez à réutiliser ce que vous venez d'apprendre sur les chaînes de caractères pour reformer le chemin du fichier que vous intéresse à partir du numéro de département…

Certaines chaînes de caractère sont interprétées comme des chaînes de caractère pour certains jeux de données (par exemple pour les codes postaux de l'Ain, à cause du "0" en début de chaîne) tandis qu'elle est interprétée comme un numérique pour les autres jeux de données. Faites en sorte que votre fonction transforme bien cette variable pour qu'elles soient toujours de classe "character" en sortie (conversion par as.character()…).

```
get_clean_data <- function(dept) {
    dept <- as.character(dept)
    print(dept)
    # Read CSV file for dept
    filename <- "data/geo_siret_XX.csv"
    filename <- str_replace(filename, "X{2}", dept)
    print(filename)
    data <- read_csv(filename)
    data <- mutate(data, Code = as.numeric(str_extract(activitePrincipaleEtablissement,
        "^....")))
    alim <- filter(data, Code %in% code_alim)
    alim <- left_join(alim, APE_Type, by = "Code")
```

```
    return(alim)
}
```

3.2 Itération

- Appelez cette fonction de manière itérative pour chacun des départements cités ci-dessus. Vous pouvez pour ce faire soit écrire une boucle for, soit utiliser la fonction map() du package purrr.

A partir des 5 jeux de données obtenus vous créerez un seul et même jeu de données alimRA_entr (données pour l'ancienne région Rhône-Alpes, où 1 ligne=1 entreprise).

- Vous pourrez si vous le souhaitez vous servir de la commande do.call("rbind",…) ou bind_rows().
- Rajoutez une variable departement (correspondant au numéro de département) à votre jeu de données alimRA_entr. Peut-être par des manipulations sur le code postal?…

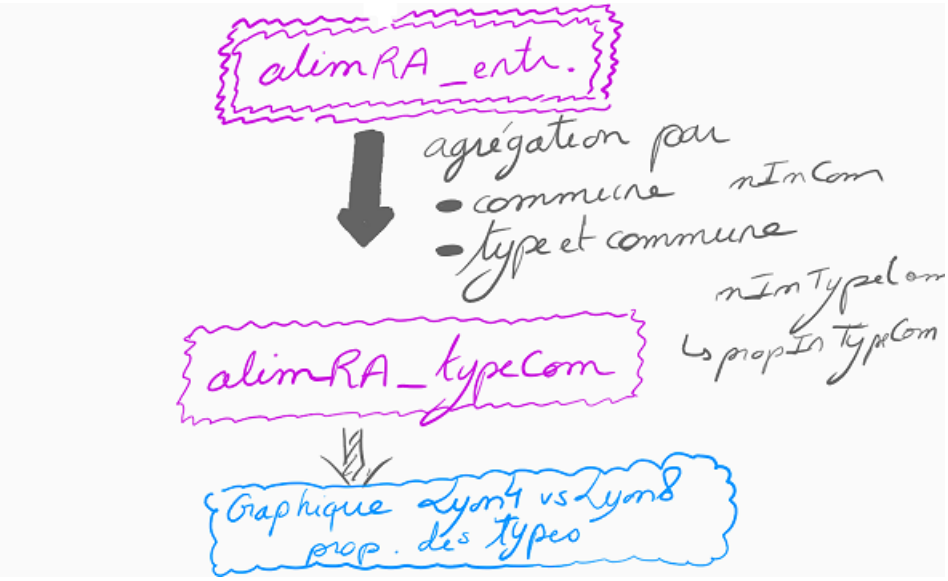
3.3 If et écriture de fichier

Vous avez dû remarquer que l'exécution de l'étape précédente prenait un peu de temps car les 5 fichiers geosirene lus sont très volumineux… En revanche la table alimRA_entr est de taille beaucoup plus raisonnable. Or, nous n'aurons besoin que de cette table pour la suite du projet. Pour éviter d'exécuter cette étape chronophage à chaque fois que vous travaillerez sur ce projet:

- exportez alimRA_entr dans un fichier alimRA_entr.csv.
- entourez la boucle for d'une structure conditionnelle if de sorte que la boucle ne soit exécutée que si le fichier alimRA_entr.csv n'existe pas (voir fonction file.exists()…)
- écrivez à la suite la commande qui servira à lire alimRA_entr.csv à chaque "tricotage" de votre rapport Rmarkdown.

---

4 Résumé par commune et type de commerce



4.1 Agrégation des données par commune et type de commerce

Agrégez la table alimRA_entr par commune et type de commerce, pour créer une table alimRA_typeCom (où une ligne correspondra à un type de commerce pour une commune):

- une variable nInCom correspondant au nombre de commerces par commune
- une variable nInTypeCom correspondant au nombre de commerces par type et commune
- une variable propInTypeCom correspondant à la proportion d'un type de commerce dans une commune

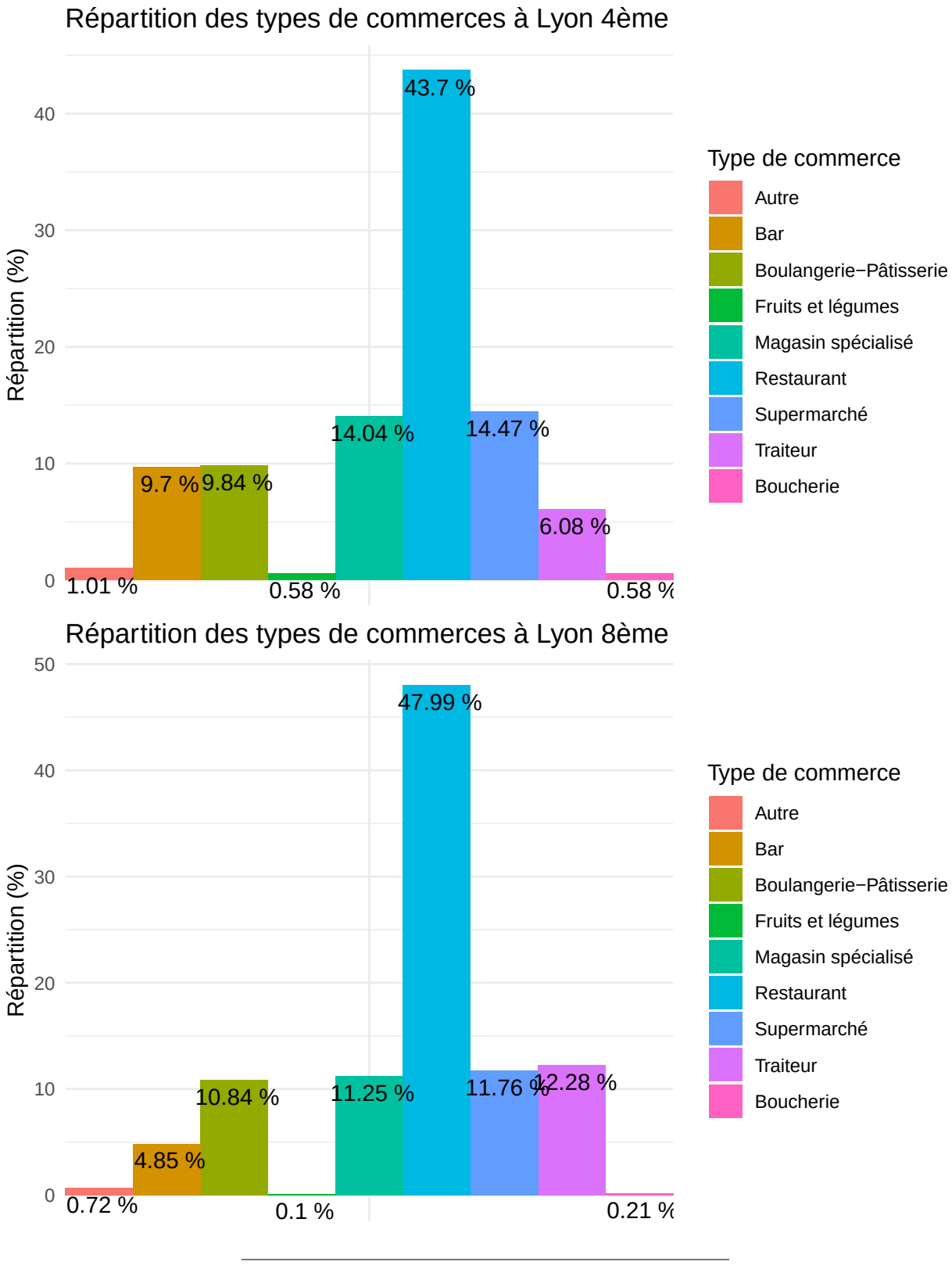
Quelles communes comptant plus de 100 commerces comptes au moins 5% de commerces de type "viande"?

Les communes comptant plus de 100 commerces, dont les commerces de type 'bar' représentent au moins 15% des commerces sont TREVOUX, BEAUREPAIRE, ROUSSILLON, SAINT-MARCELLIN, CHAMROUSSE, CHARLIEU, FIRMINY, ROANNE, BRIOUDE, LANGEAC, THIZY-LES-BOURGS.

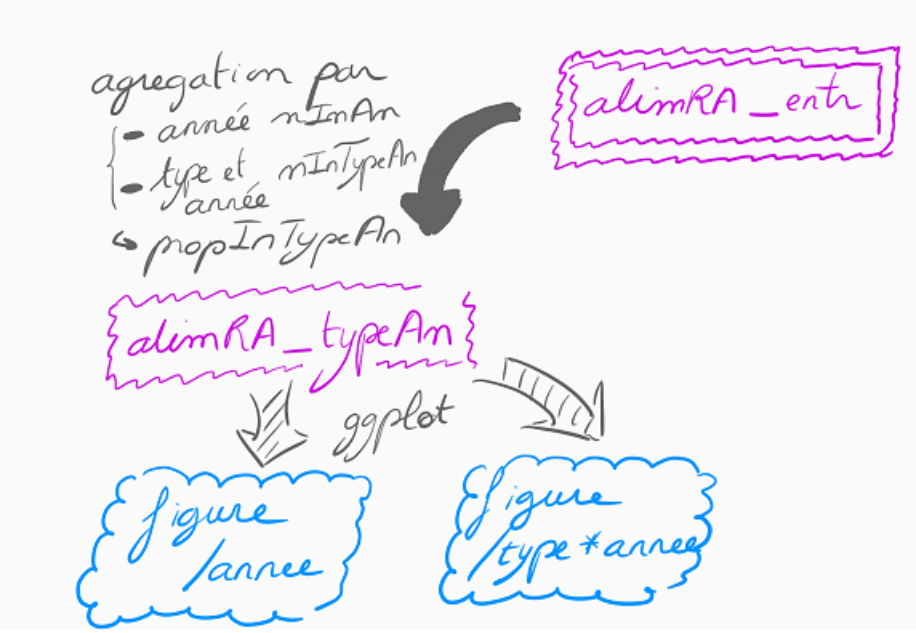
Les communes comptant plus de 100 commerces, dont les commerces de type 'viande' représentent au moins 5% des commerces sont CHARLIEU, FEURS, MONTBRISON, LA TALAUDIERE, YSSINGEAUX, CORBAS.

4.2 Graphique

- Réalisez un graphique montrant les proportions des différents types de commerces pour LYON 4EME et LYON 8EME.


Répartition des types de commerces à Lyon 4ème
Répartition (%)
0
10
20
30
40
1.01 %
9.7 %
9.84 %
0.58 %
14.04 %
43.7 %
14.47 %
6.08 %
0.58 %
Type de commerce
Autre
Bar
Boulangerie−Pâtisserie
Fruits et légumes
Magasin spécialisé
Restaurant
Supermarché
Traiteur
Boucherie
Répartition des types de commerces à Lyon 8ème
Répartition (%)
0
10
20
30
40
50
0.72 %
4.85 %
10.84 %
0.1 %
11.25 %
47.99 %
11.76 %
12.28 %
0.21 %
Type de commerce
Autre
Bar
Boulangerie−Pâtisserie
Fruits et légumes
Magasin spécialisé
Restaurant
Supermarché
Traiteur
Boucherie

5 Evolution dans le temps des créations d'entreprise



5.1 Manipuler des dates avec lubridate

Nous allons nous intéresser aux dates de création des entreprises de notre base alimRA_entr (variable dateCreationEtablissement).

Pour le moment, dateCreationEtablissement est considéré comme une variable de type "chaîne de caractères". Pour faire comprendre à R qu'il s'agit en réalité d'une date (et lui faire comprendre comment elle est mise en forme) nous allons faire appel au package lubridate.

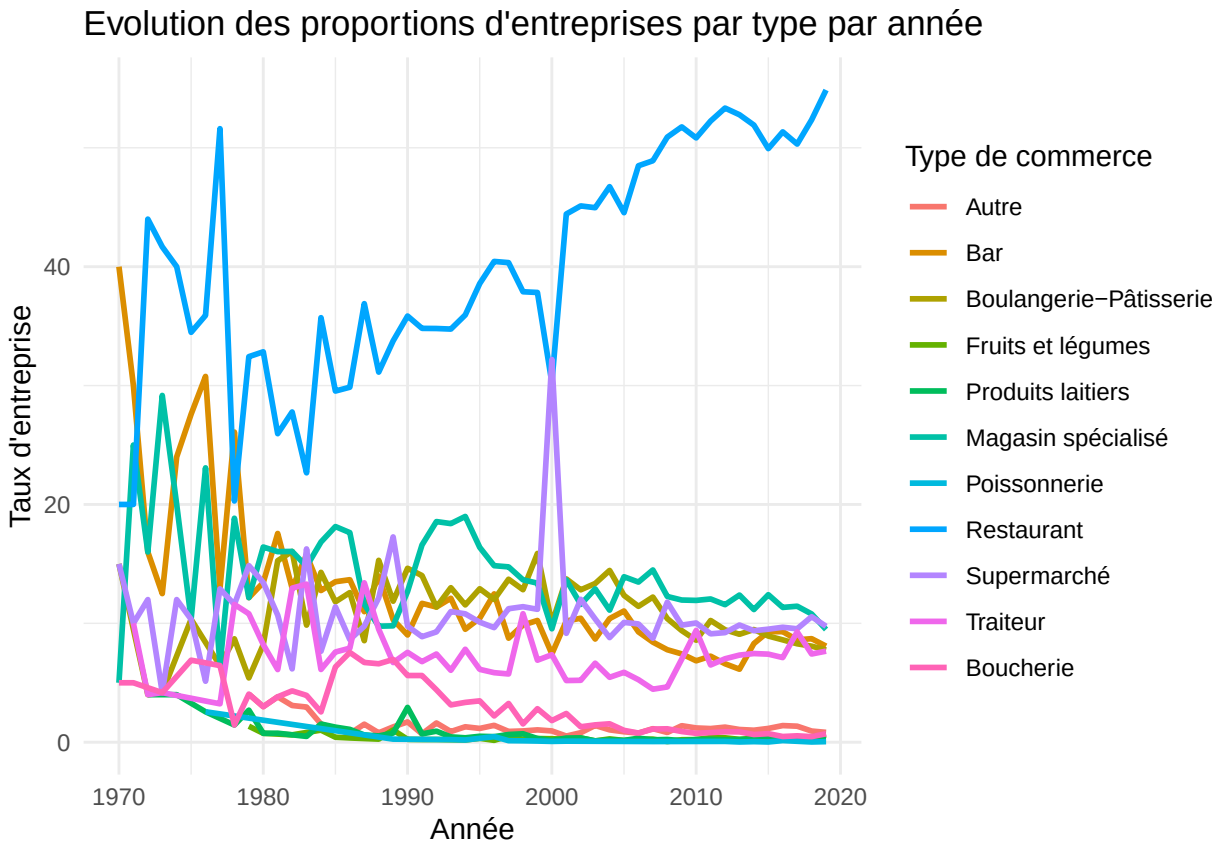
- Installez et chargez le package lubridate.
- Transformez le tableau alimRA_entr en modifiant la classe de dateCreationEtablissement à l'aide d'une fonction de lubridate.
- Ajoutez une variable annee au tableau alimRA_entr à l'aide, à nouveau, d'une des fonctions de lubridate.

5.2 Résumé, filtre

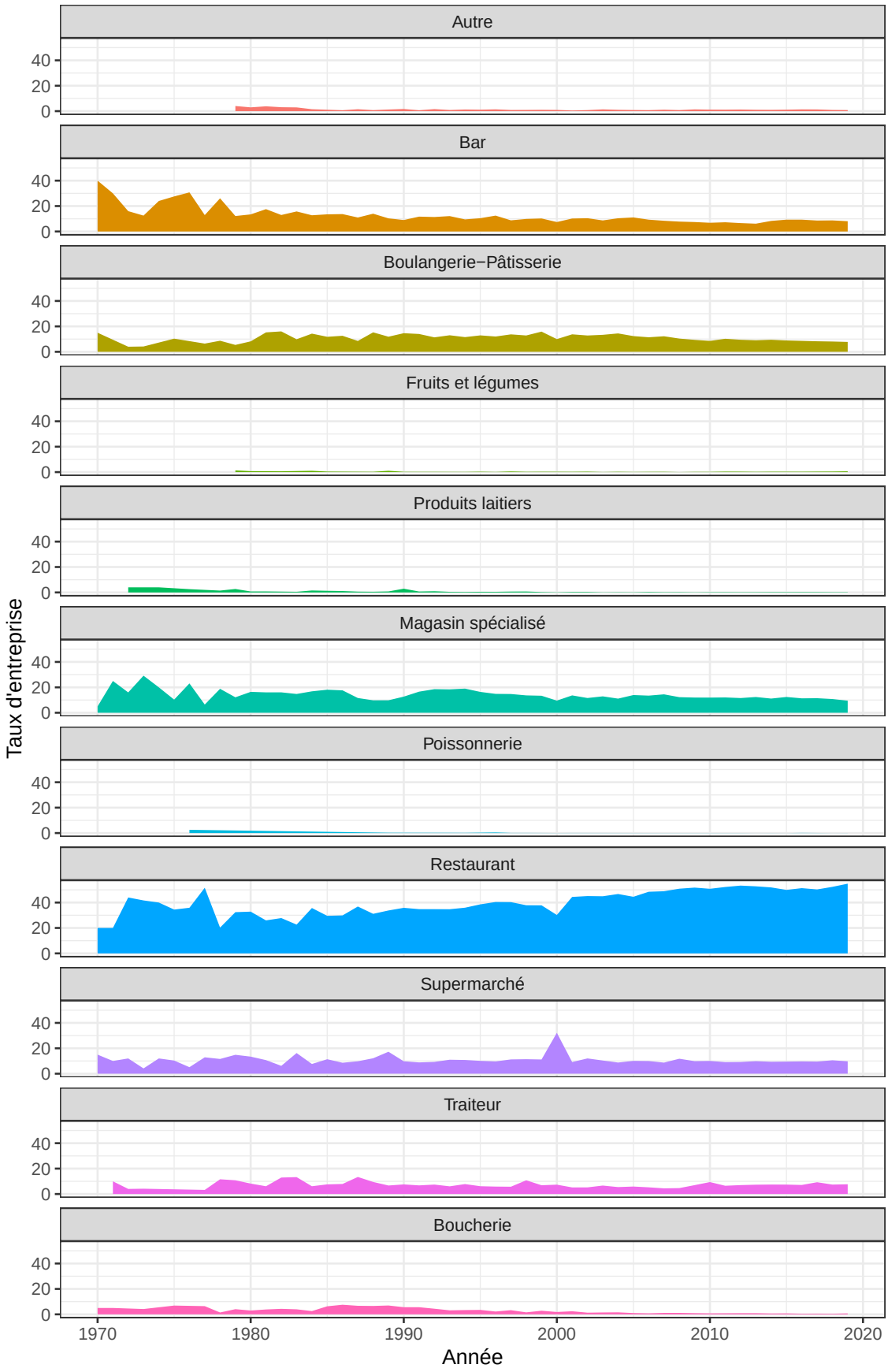
- Créez une table alimRA_typeAn qui recense le nombre d'entreprises par année (nInAn), et par type*année (nInTypeAn).
- Filtrez les données de alimRA_typeAn pour ne garder que les entreprises dont la création correspond aux années >=1970.

5.3 Graphiques: évolution du nombre d'entreprises au cours du temps

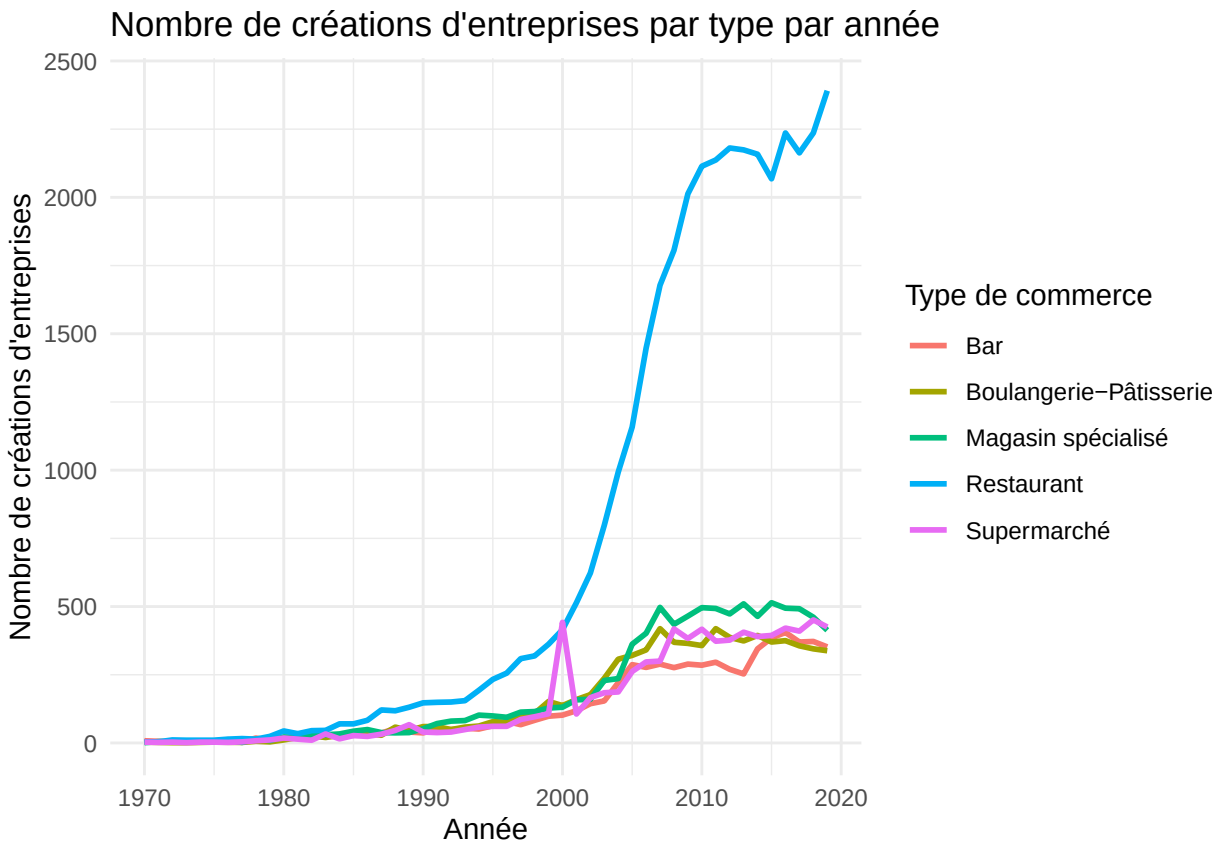
- Installez et chargez le package ggplot2
- Réalisez un graphique représentant l'évolution des proportions d'entreprises (par type) par année.



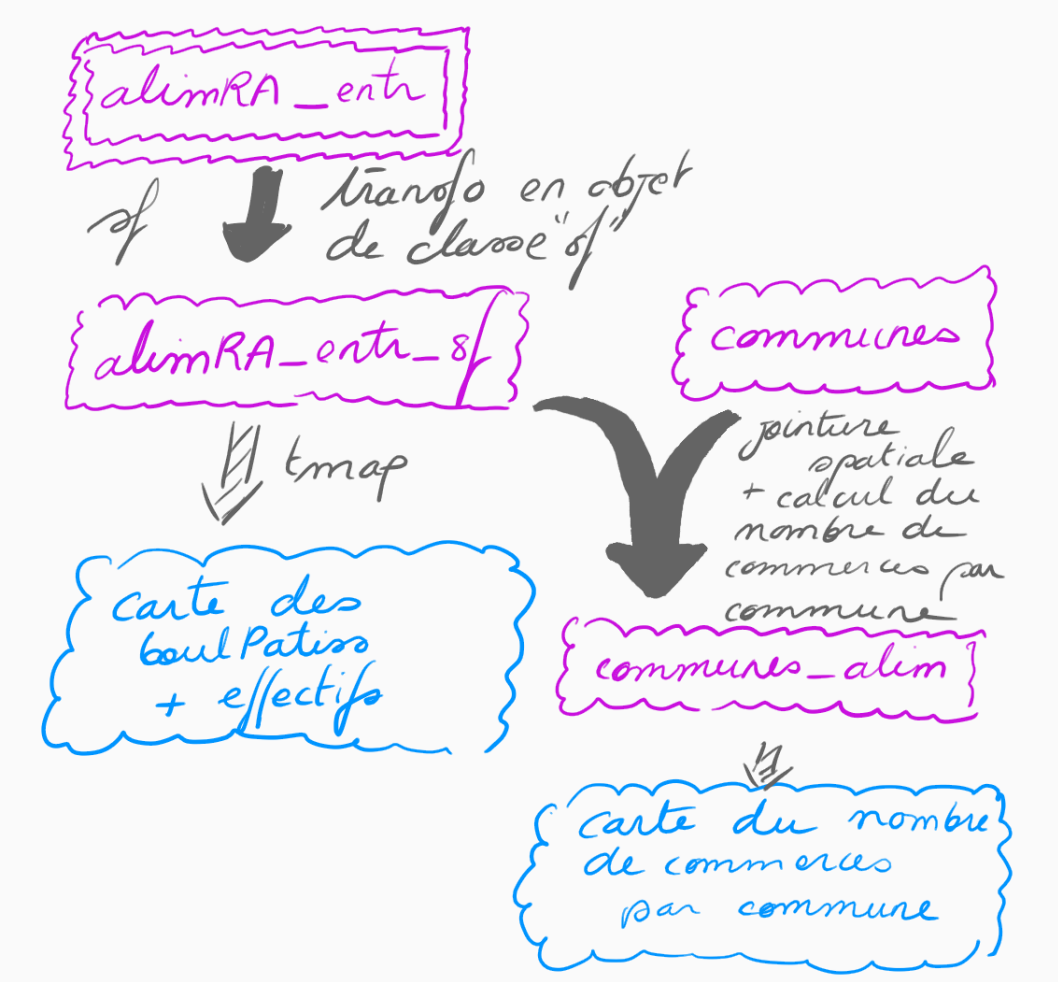
On peut aussi séparer les types de commerce et créer une grille avec un graphique pour chaque type :


Autre
Bar
Boulangerie−Pâtisserie
Fruits et légumes
Produits laitiers
Magasin spécialisé
Poissonnerie
Restaurant
Supermarché
Traiteur
Boucherie
Taux d'entreprise
0
20
40
1970
1980
1990
2000
2010
2020
Année

- Réalisez ce même graphique, mais en représentant le nombre de créations d'entreprises par année et par type, pour les 5 types comptant le plus de créations d'entreprises (au total).

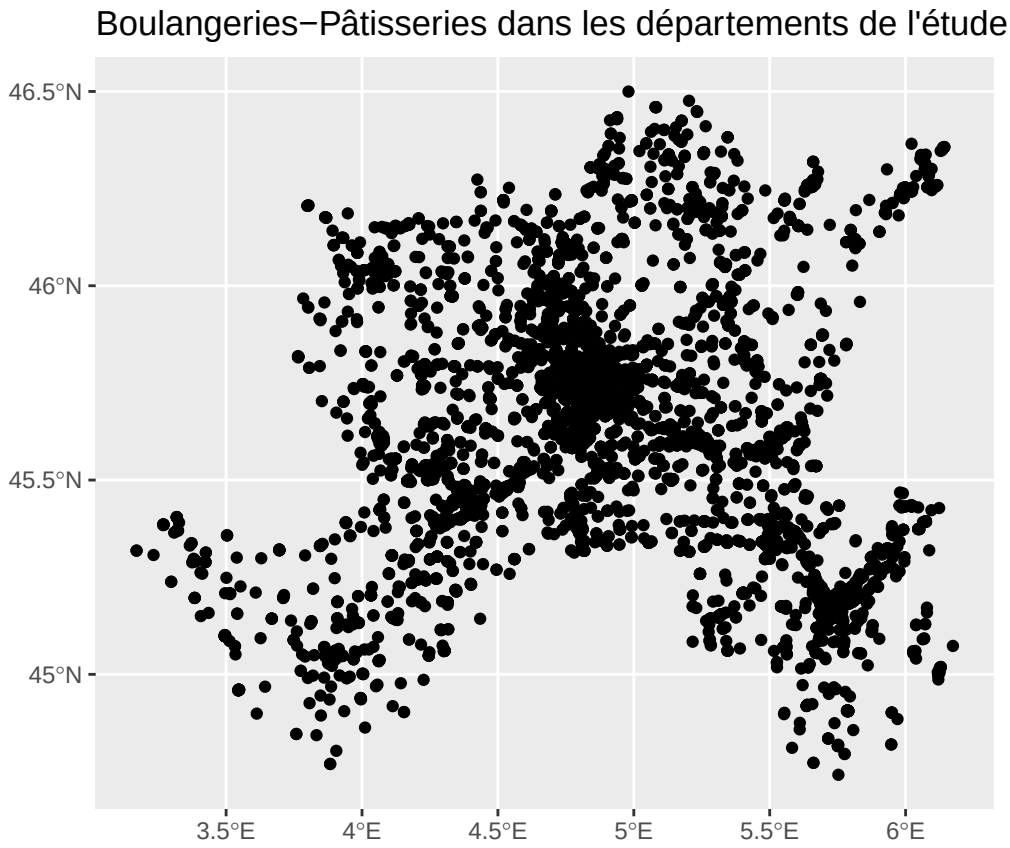


6 Cartes

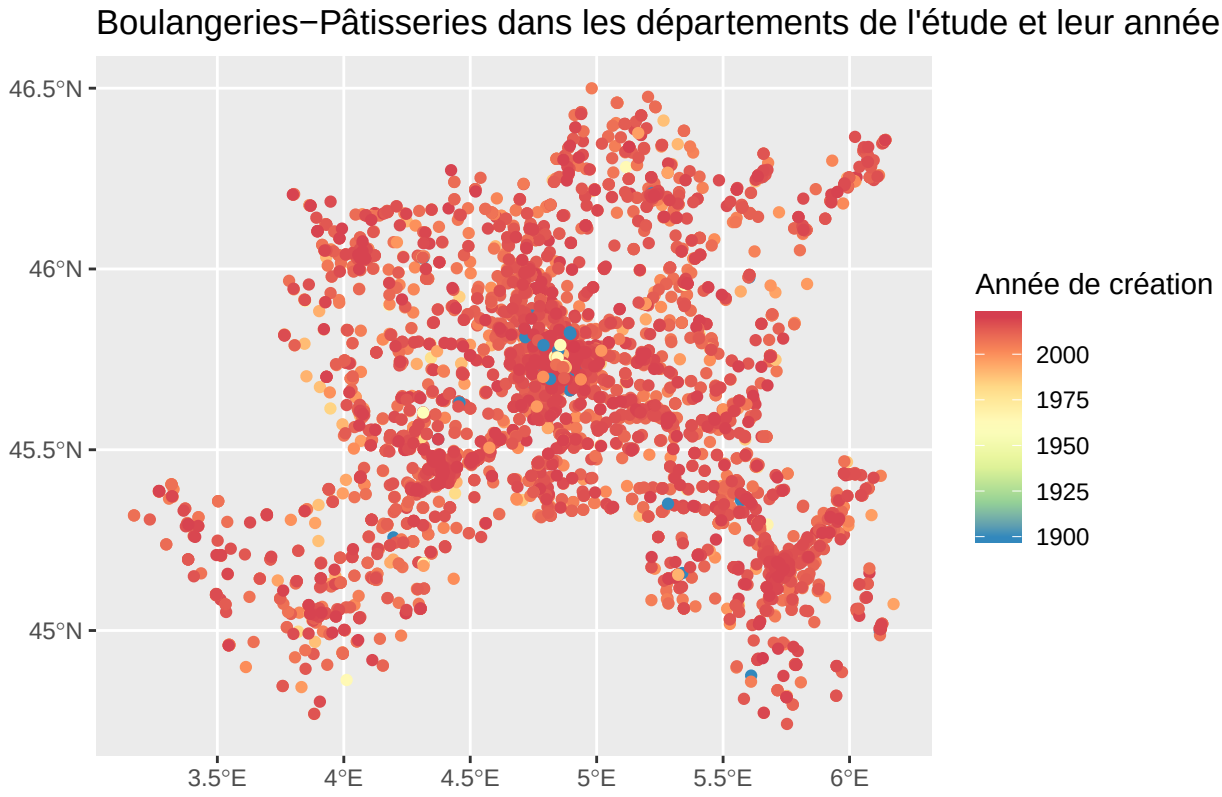


6.1 Carte des boulangeries-pâtisseries

- Repartez de la table alim_entr pour en faire un objet de classe “sf”. Vous vous servirez pour cela des colonnes “longitude” et “latitude” et exclurez les entreprises pour lesquelles ces colonnes ne sont pas renseignées.
- Réalisez une carte montrant le semis de points correspondant aux boulangeries-pâtisseries.



- Essayez de représenter à travers cette carte l'année de création de l'entreprise (de la manière qui vous semblera la plus pertinente).



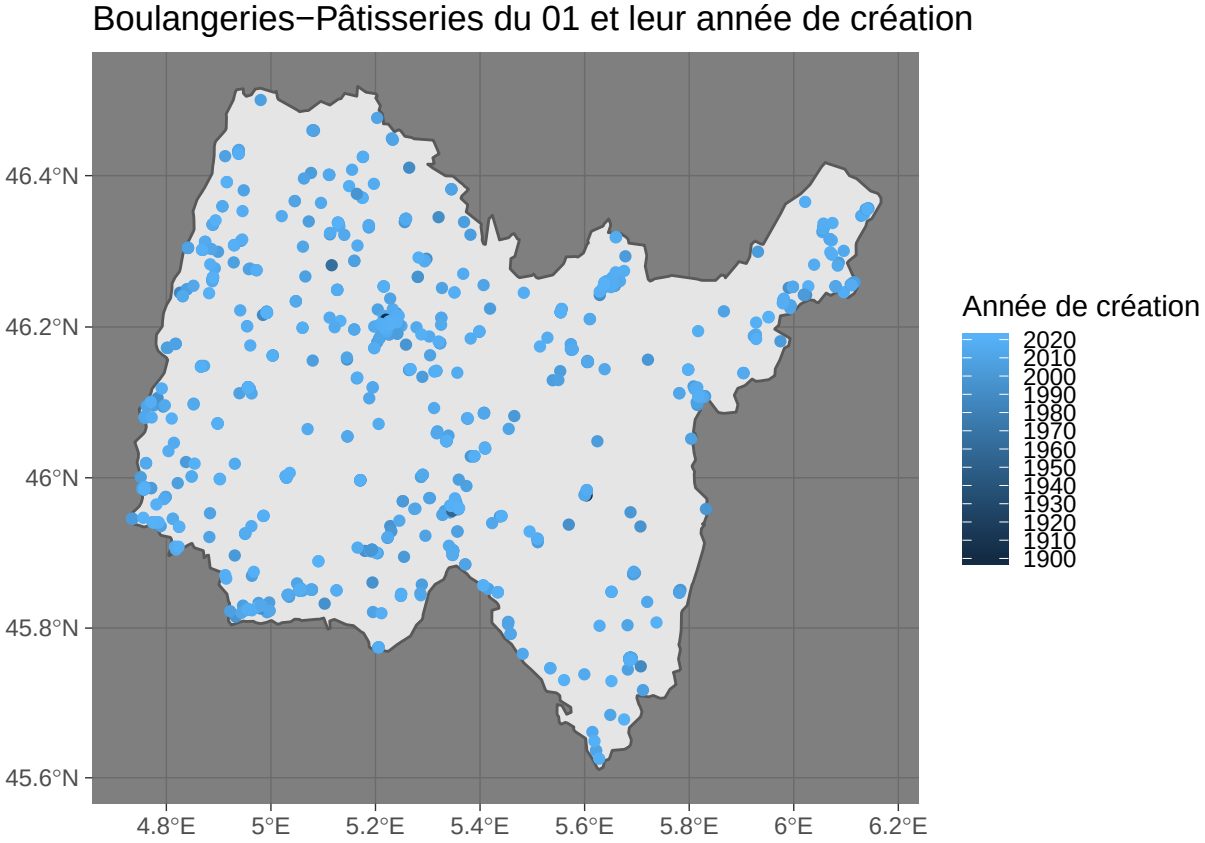
6.2 Carte des proportions de commerce par commune

- Téléchargez le shapefile des limites de communes en France et filtrez pour ne garder que les départements considérés ci-dessus.

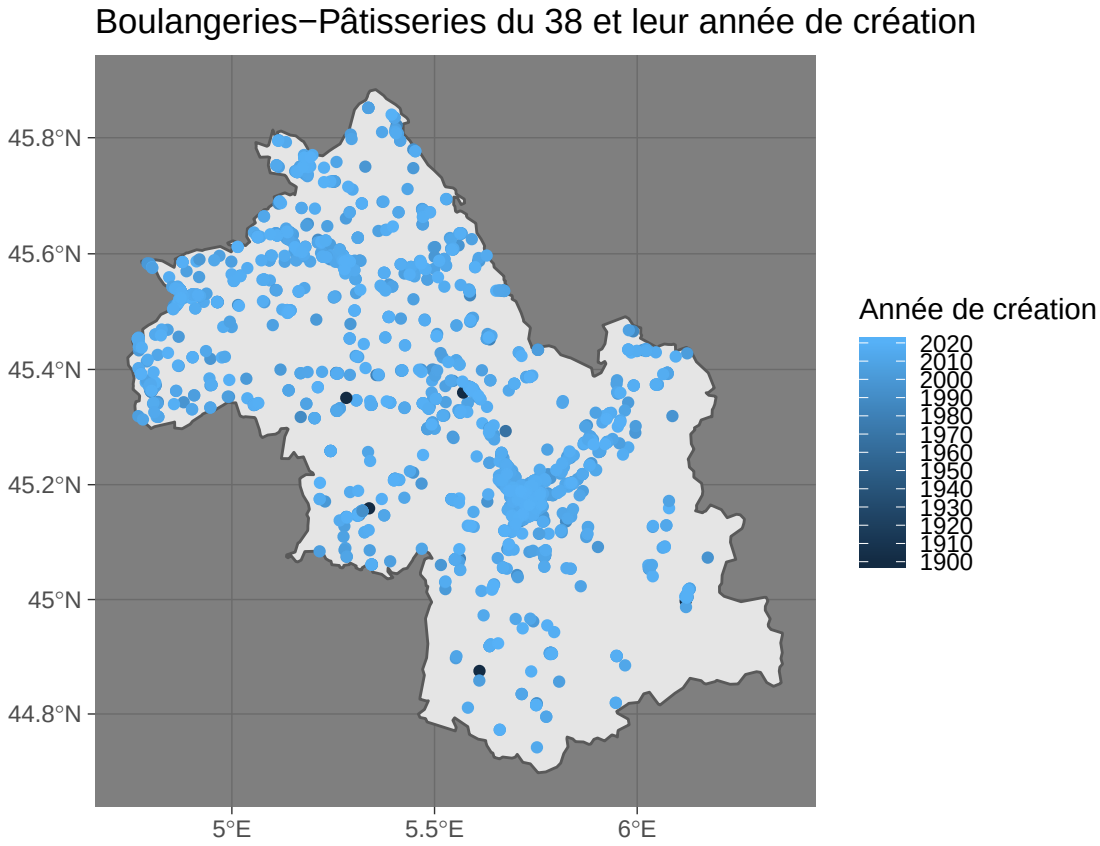
On crée une fonction pour afficher les commerces d'un département

On peut afficher les boulangeries-pâtisseries avec leur année de création, par département :

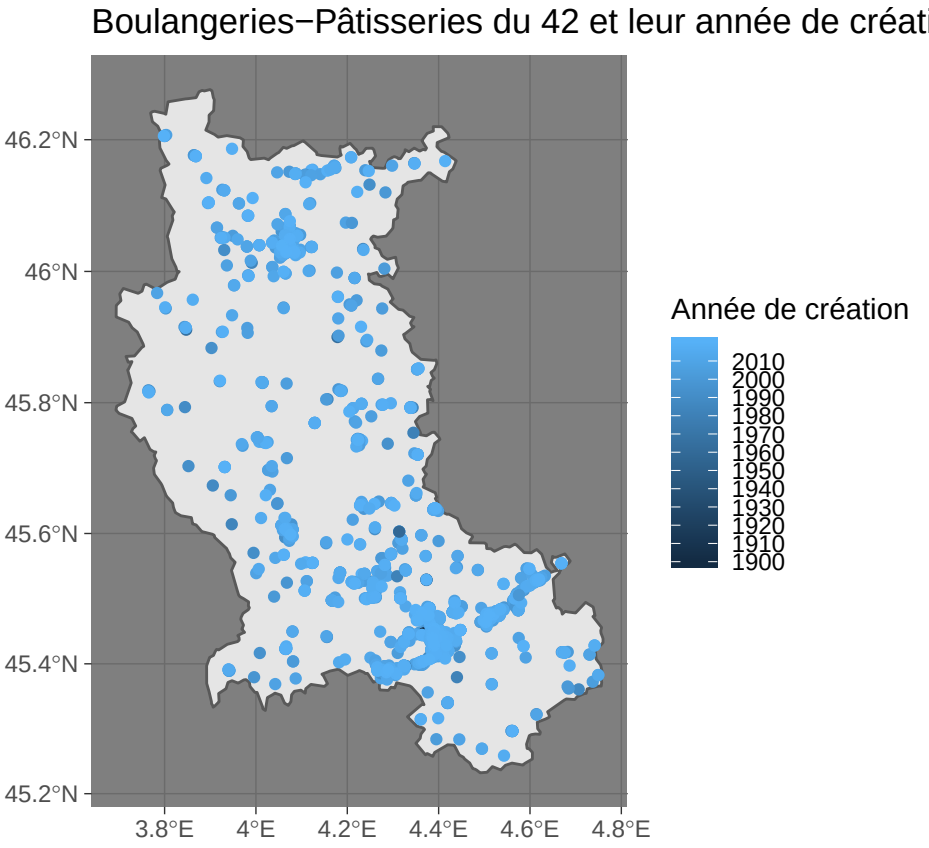
```
## [[1]]
```



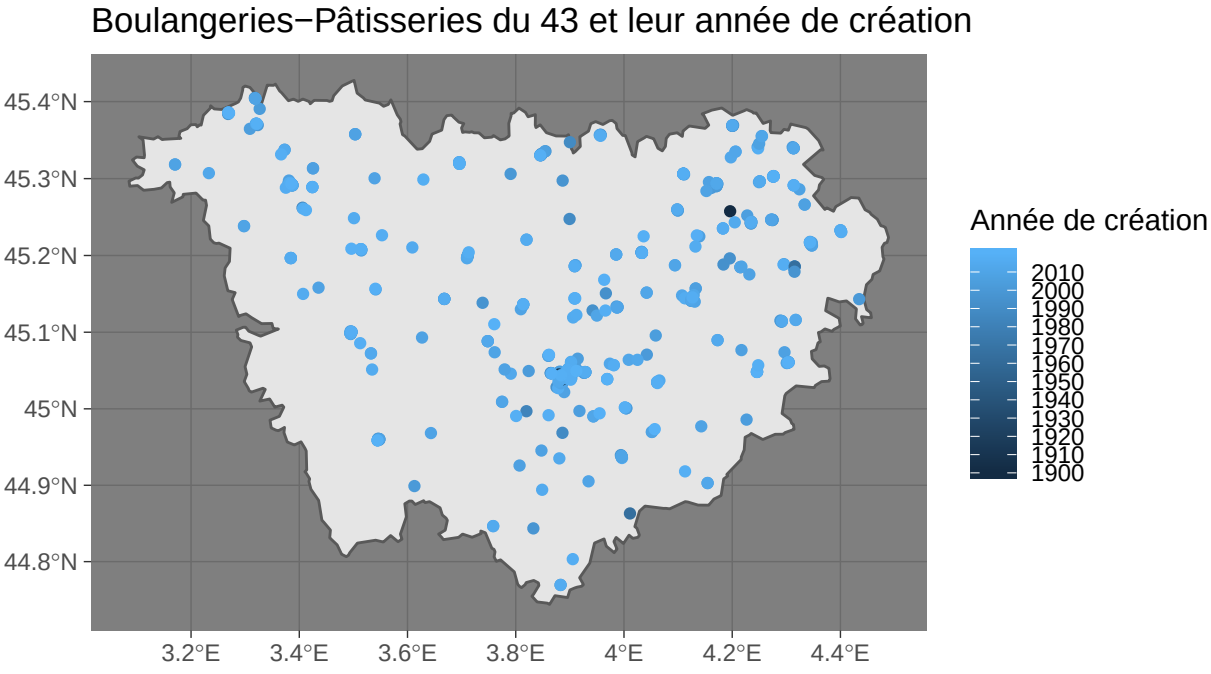

```
##
## [[2]]
```


Boulangeries−Pâtisseries du 38 et leur année de création
45.8°N
45.6°N
45.4°N
45.2°N
45°N
44.8°N
5°E
5.5°E
6°E
Année de création
2020
2010
2000
1990
1980
1970
1960
1950
1940
1930
1920
1910
1900

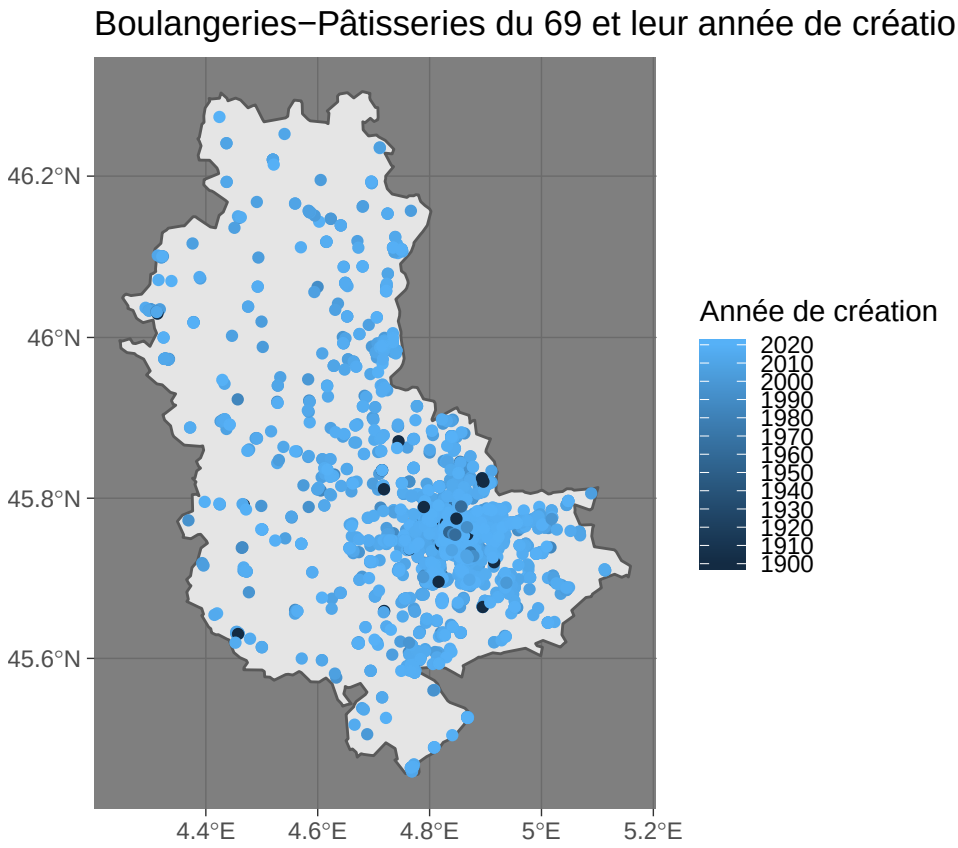
```
##
## [[3]]
```


Boulangeries−Pâtisseries du 42 et leur année de création
46.2°N
46°N
45.8°N
45.6°N
45.4°N
45.2°N
3.8°E
4°E
4.2°E
4.4°E
4.6°E
4.8°E
Année de création
2010
2000
1990
1980
1970
1960
1950
1940
1930
1920
1910
1900

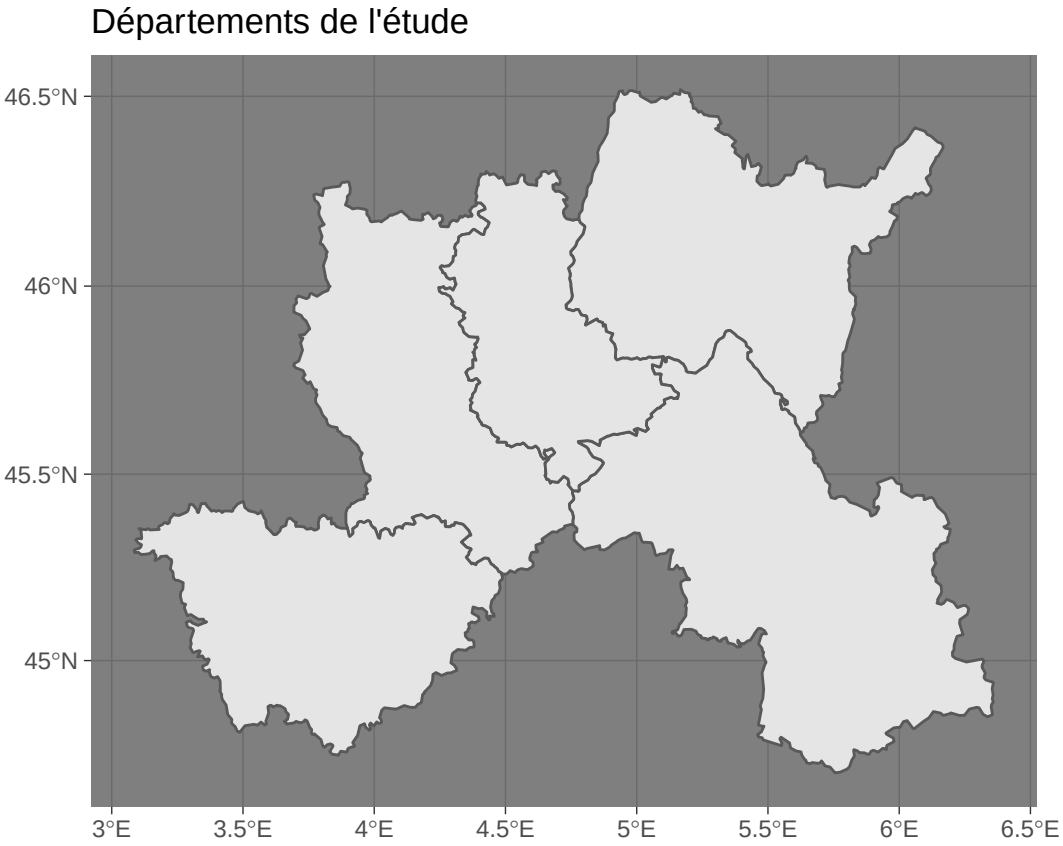
```
##
## [[4]]
```


Boulangeries−Pâtisseries du 43 et leur année de création
45.4°N
45.3°N
45.2°N
45.1°N
45°N
44.9°N
44.8°N
3.2°E
3.4°E
3.6°E
3.8°E
4°E
4.2°E
4.4°E
Année de création
2010
2000
1990
1980
1970
1960
1950
1940
1930
1920
1910
1900

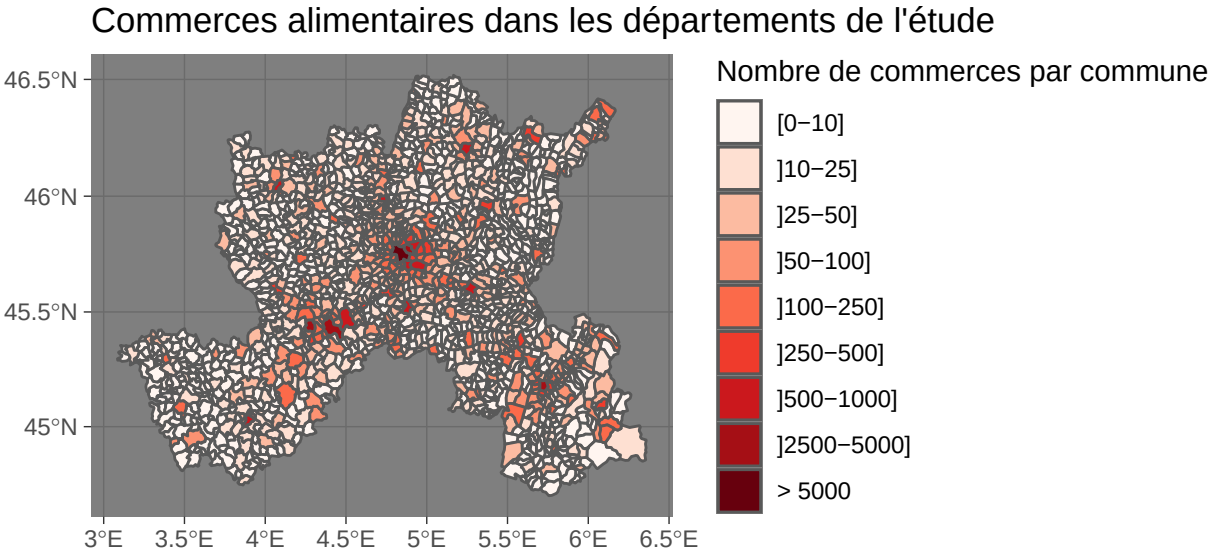
```
##
## [[5]]
```



On affiche maintenant les départements de l'étude à partir du fichier SHP récupéré précédemment :



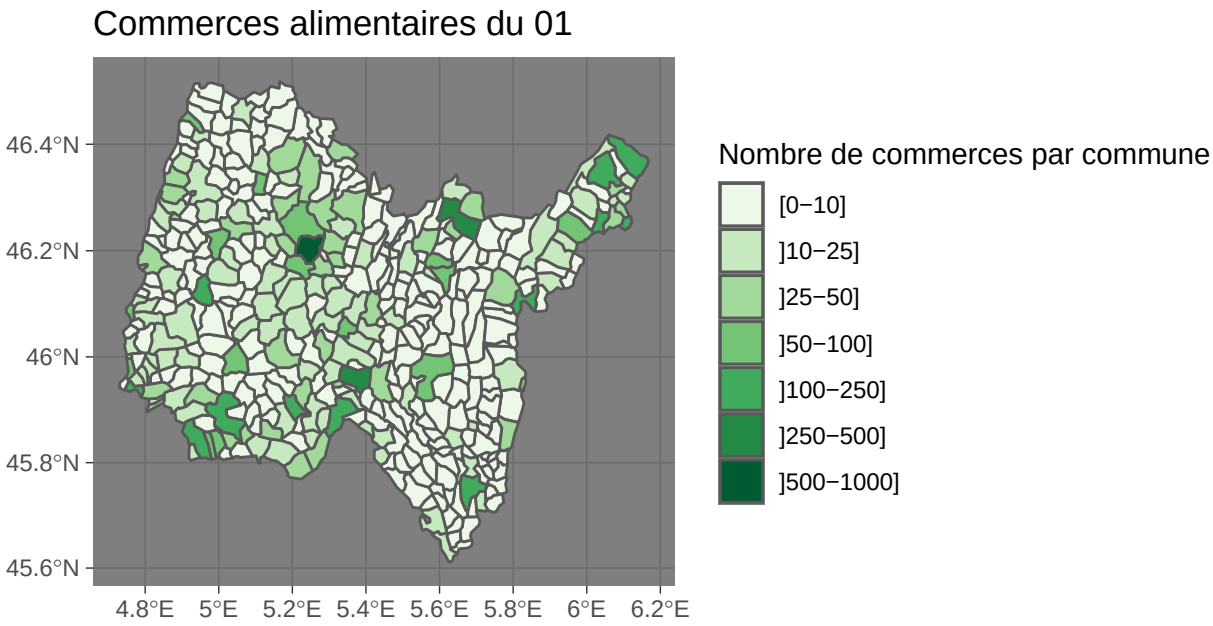
- Joignez aux communes les informations concernant les commerces (st_join()…) et calculez le nombre de commerces par commune.
- Produisez une carte montrant le nombre de commerces par commune. Vous aurez sans doute à retravailler l'échelle colorée…



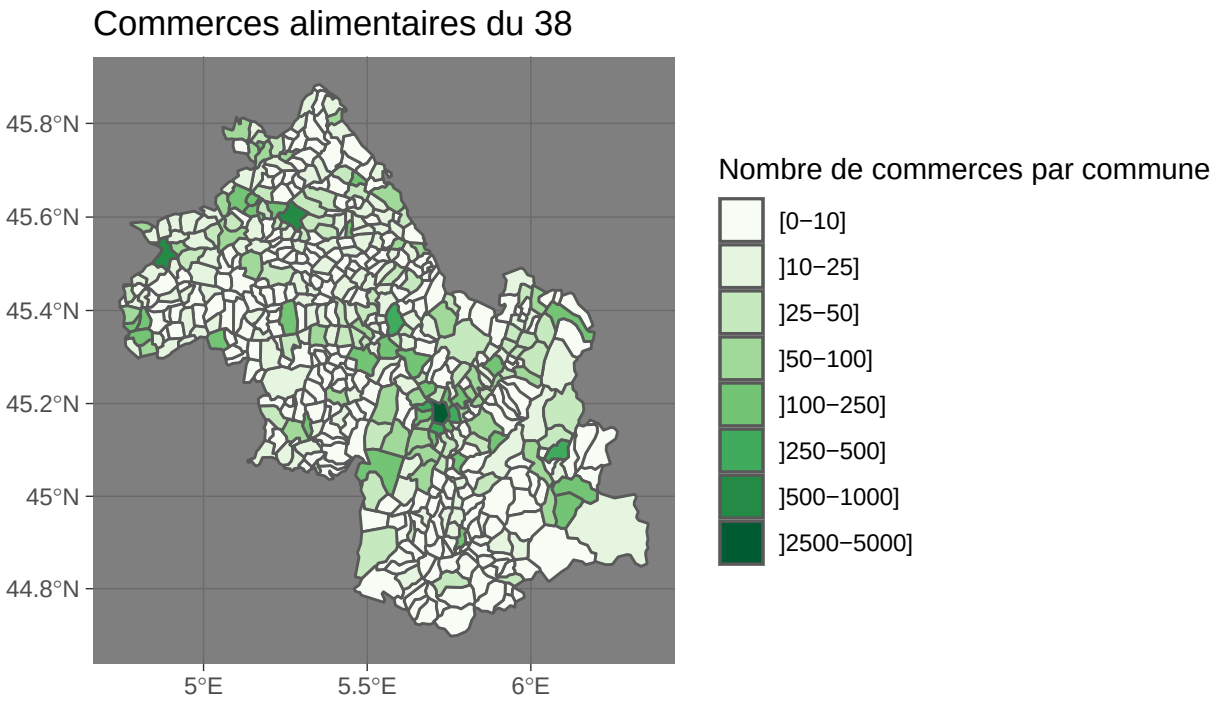
On crée une fonction pour afficher le nombre de commerces par commune d'un département.

Puis on affiche les résultats :

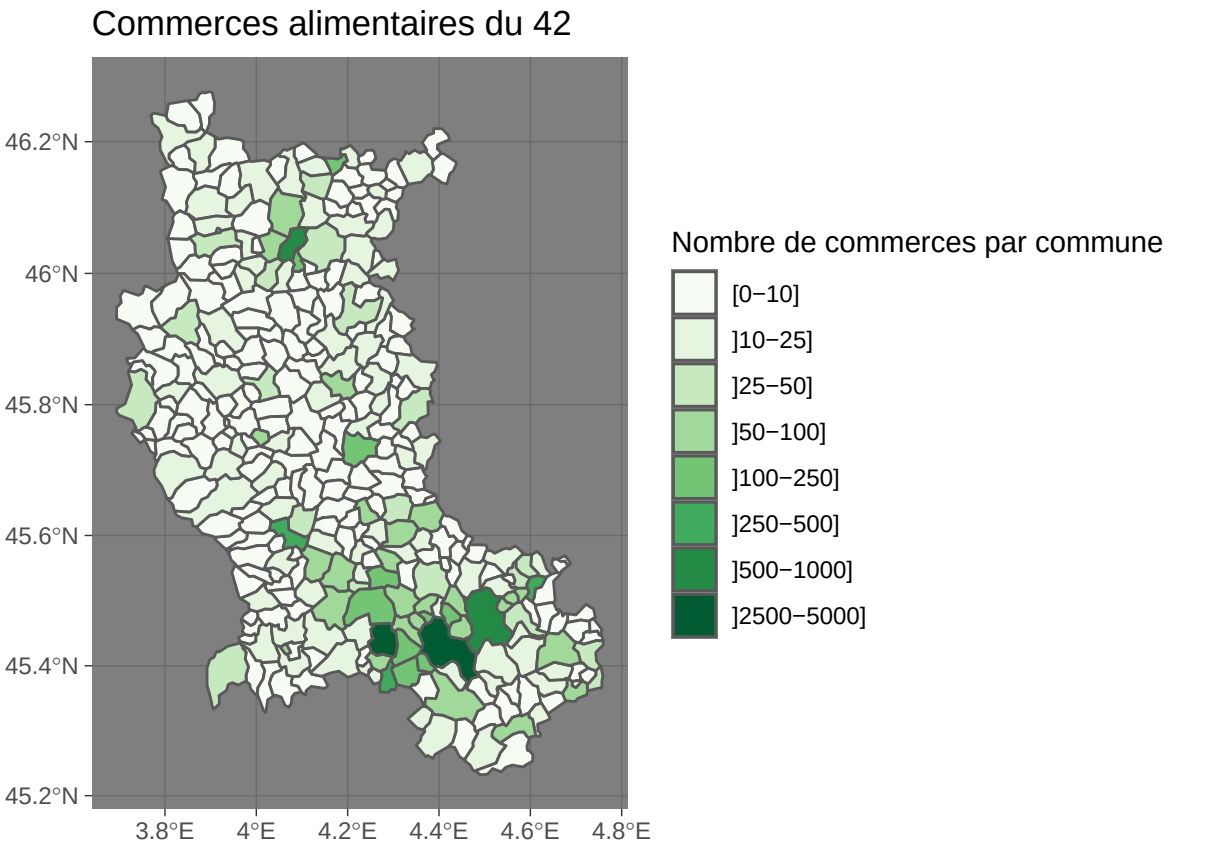
```
## [[1]]
```


Commerces alimentaires du 01
46.4°N
46.2°N
46°N
45.8°N
45.6°N
4.8°E
5°E
5.2°E
5.4°E
5.6°E
5.8°E
6°E
6.2°E
Nombre de commerces par commune
[0−10]
]10−25]
]25−50]
]50−100]
]100−250]
]250−500]
]500−1000]

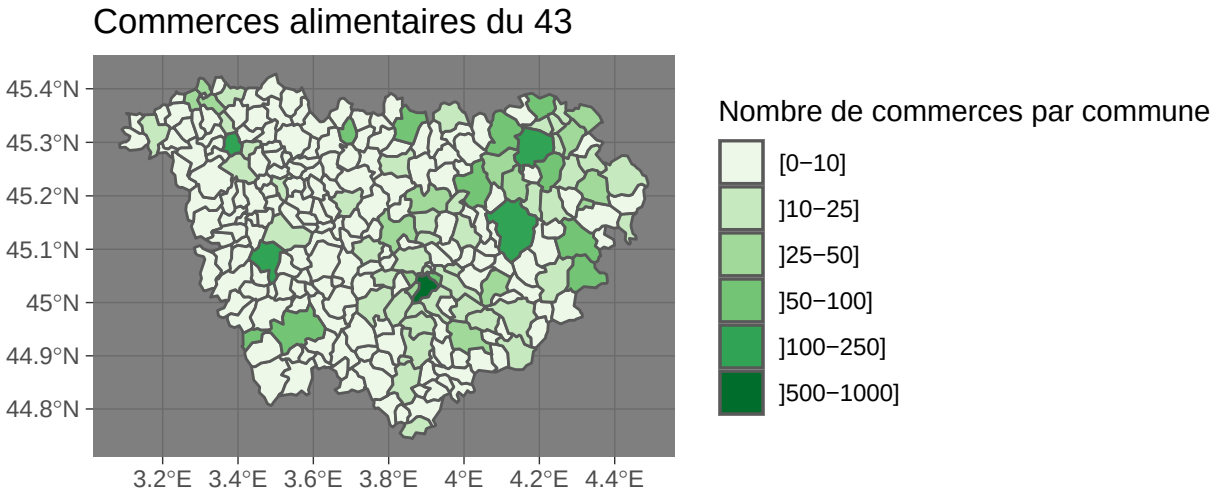
```
##
## [[2]]
```


Commerces alimentaires du 38
45.8°N
45.6°N
45.4°N
45.2°N
45°N
44.8°N
5°E
5.5°E
6°E
Nombre de commerces par commune
[0−10]
]10−25]
]25−50]
]50−100]
]100−250]
]250−500]
]500−1000]
]2500−5000]

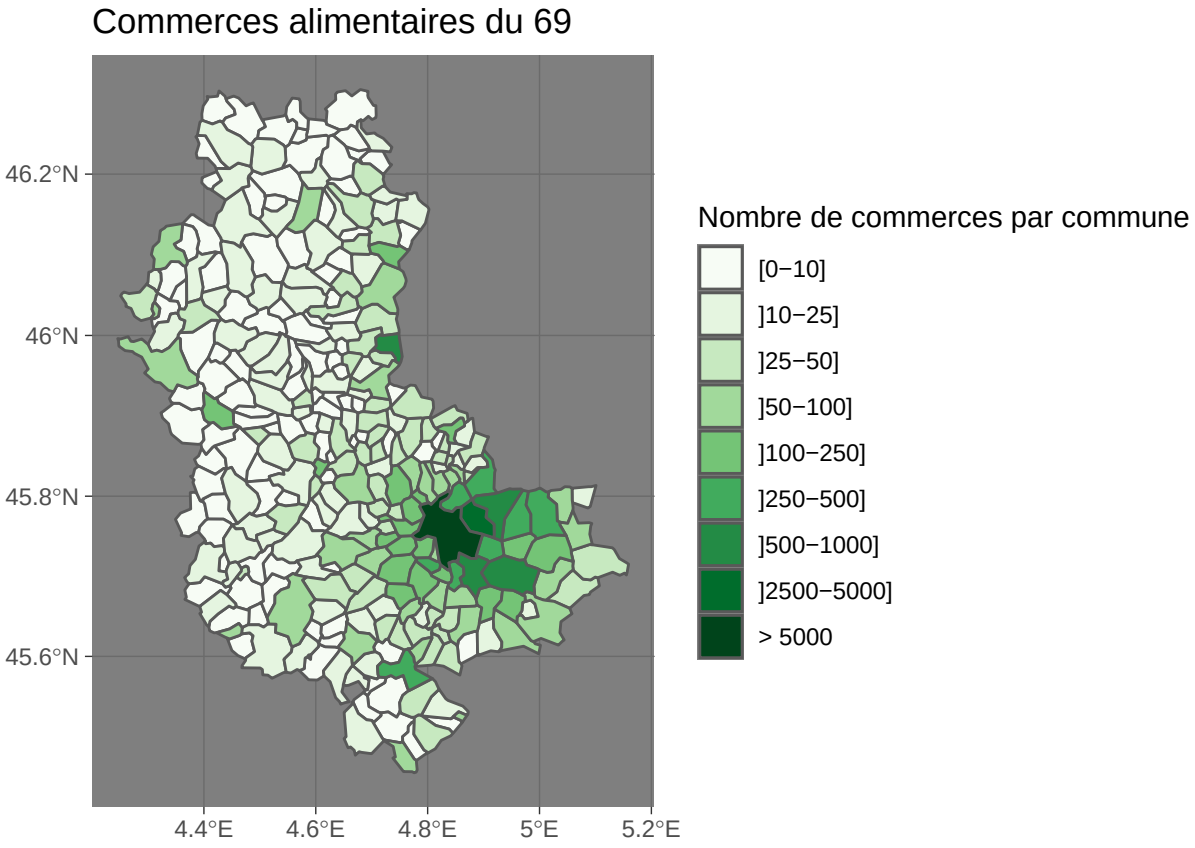
```
##
## [[3]]
```


Commerces alimentaires du 42
46.2°N
46°N
45.8°N
45.6°N
45.4°N
45.2°N
3.8°E
4°E
4.2°E
4.4°E
4.6°E
4.8°E
Nombre de commerces par commune
[0−10]
]10−25]
]25−50]
]50−100]
]100−250]
]250−500]
]500−1000]
]2500−5000]

```
##
## [[4]]
```


Commerces alimentaires du 43
45.4°N
45.3°N
45.2°N
45.1°N
45°N
44.9°N
44.8°N
3.2°E
3.4°E
3.6°E
3.8°E
4°E
4.2°E
4.4°E
Nombre de commerces par commune
[0−10]
]10−25]
]25−50]
]50−100]
]100−250]
]500−1000]

```
##
## [[5]]
```



7 Session Info

Pour finir voici les informations de la session utilisée pour ce TP :

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Catalina 10.15.2
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] fr_FR.UTF-8/fr_FR.UTF-8/fr_FR.UTF-8/C/fr_FR.UTF-8/fr_FR.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] rmdformats_0.3.6 prettydoc_0.3.1  plotly_4.9.1     sf_0.8-0
##  [5] ggplot2_3.2.1    tidyr_1.0.0      scales_1.0.0     lubridate_1.7.4
##  [9] purrr_0.3.3      stringr_1.4.0    readr_1.3.1      dplyr_0.8.3
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.5   xfun_0.10          colorspace_1.4-1
##  [4] vctrs_0.2.0        htmltools_0.3.6    viridisLite_0.3.0
##  [7] yaml_2.2.0         utf8_1.1.4         rlang_0.4.2
```

```
## [10] e1071_1.7-2        pillar_1.4.2       glue_1.3.1
## [13] withr_2.1.2        DBI_1.0.0          RColorBrewer_1.1-2
## [16] lifecycle_0.1.0    munsell_0.5.0      gtable_0.3.0
## [19] htmlwidgets_1.3    evaluate_0.14      labeling_0.3
## [22] knitr_1.25         class_7.3-15       fansi_0.4.0
## [25] Rcpp_1.0.2         KernSmooth_2.23-15 backports_1.1.4
## [28] classInt_0.4-1     formatR_1.7        jsonlite_1.6
## [31] hms_0.5.2          digest_0.6.21      stringi_1.4.3
## [34] bookdown_0.16      grid_3.6.1         cli_1.1.0
## [37] tools_3.6.1        magrittr_1.5       lazyeval_0.2.2
## [40] tibble_2.1.3       crayon_1.3.4       pkgconfig_2.0.3
## [43] zeallot_0.1.0      ellipsis_0.3.0     data.table_1.12.2
## [46] assertthat_0.2.1   rmarkdown_2.0      httr_1.4.1
## [49] R6_2.4.0           units_0.6-4        compiler_3.6.1
```