PipelineViz

François Pelletier

Analyse d'un jeu de données ouvertes

Source: Données sur les évenements de pipeline de Janvier 2004 au présent

Chargement du dictionnaire de données

```
# install.packages(c("data.table", "dplyr","maps","ggmap"))
options(unzip = 'internal')
library("data.table")
library("dplyr")
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:data.table':

    between, first, last

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
library("maps")
library("ggmap")
```

```
Loading required package: ggplot2
```

```
i Google's Terms of Service: <https://mapsplatform.google.com>
  Stadia Maps' Terms of Service: <https://stadiamaps.com/terms-of-service>
  OpenStreetMap's Tile Usage Policy: <https://operations.osmfoundation.org/policies/tiles>
i Please cite ggmap if you use it! Use `citation("ggmap")` for details.
```

```r
library("reshape2")
```

```
Attaching package: 'reshape2'

The following objects are masked from 'package:data.table':

    dcast, melt
```

```r
library("lubridate")
```

```
Attaching package: 'lubridate'

The following objects are masked from 'package:data.table':

    hour, isoweek, isoyear, mday, minute, month, quarter, second, wday,
    week, yday, year

The following objects are masked from 'package:base':

    date, intersect, setdiff, union
```

```r
library("rpart")
library("rpart.plot")
```

Lecture des données

```
datadict <- read.csv(file = "dd-20150216.csv",header = TRUE, stringsAsFactors = TRUE, encodi
  as.data.table()
donnees <- read.csv(file = "p2015-07-fr.csv",header = TRUE,stringsAsFactors = TRUE) %>%
  as.data.table()

class_donnees <- sapply(donnees,class)
character_columns <- names(donnees)[class_donnees == "character"]
num_columns <- names(donnees)[class_donnees != "character"]

donnees_conv <- cbind(donnees[, lapply(.SD, iconv, "utf-8", "latin1"),
                   .SDcols = character_columns],donnees[,.SD,.SDcols = num_columns])
```

Geocoder les incidents

```
LocationsUnique <- unique(donnees_conv[,.(LOCATION,PROVINCE)])
catLocationsUnique <- unique(paste(LocationsUnique$LOCATION, ifelse(!is.na(LocationsUnique$PR
geocodedLocations <- cbind(LocationsUnique,geocode(catLocationsUnique))
save(geocodedLocations, file = "geocodedLocations.Rdata")
```

Joindre les données géocodées aux données existantes

```
load("geocodedLocations.Rdata")
donnees_conv2 <-as.data.table(merge(donnees_conv,geocodedLocations, by=c("LOCATION","PROVINCE
donnees_conv3 <- donnees_conv2[!is.na(PROVINCE) & !is.na(LOCATION)]
```

Préparation de la carte de fond

```
range_lat <- range(donnees_conv3$lat)
range_lon <- range(donnees_conv3$lon)

openstreetmap1 <- get_openstreetmap(bbox = c(left = range_lon[1],
                                             bottom = range_lat[1],
                                             right = range_lon[2],
                                             top = range_lat[2]),
    scale = 32500000,
    format = c("png"),
```

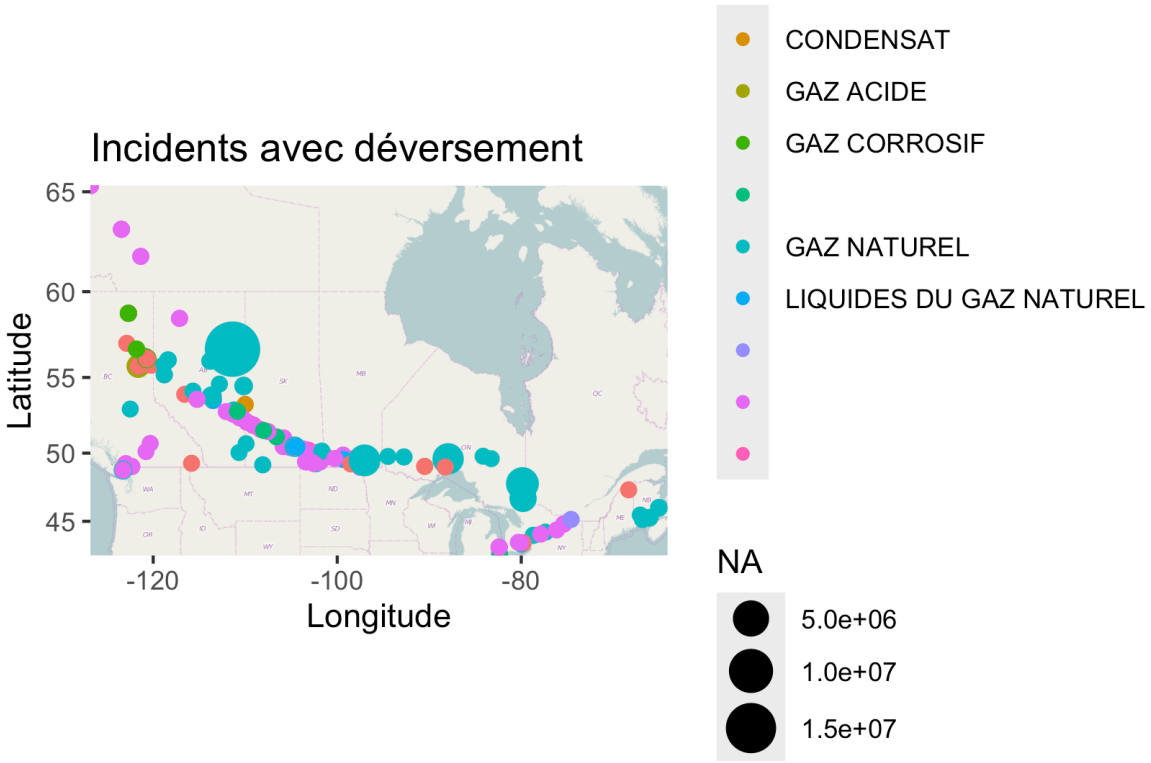
```
    crop = TRUE, messaging = FALSE, urlonly = FALSE, color = c("color", "bw"))

save(openstreetmap1, file = "openstreetmap1.Rdata")
```

Chargement de la carte de fond

```
load("openstreetmap1.Rdata")
```

Carte des incidents avec déversement

```
donnees_incidents <- donnees_conv3[QUANTITY_LOST > 0]
ggmap(openstreetmap1) +
  geom_point(aes(x = lon,
                 y = lat,
                 colour = PRODUCT_TYPE,
                 size = QUANTITY_LOST),
             data = donnees_incidents) +
  scale_size(range = c(2, 8), name=datadict[datadict$Nom.de.colonne.PODS=="QUANTITY_LOST"][2]
  scale_colour_discrete(name=datadict[datadict$Nom.de.colonne.PODS=="PRODUCT_TYPE"][2]) +
  ggtitle(label = "Incidents avec déversement") +
  xlab("Longitude") +
  ylab("Latitude")
```



initiales

Cette fonction extrait les initiales d'un mot

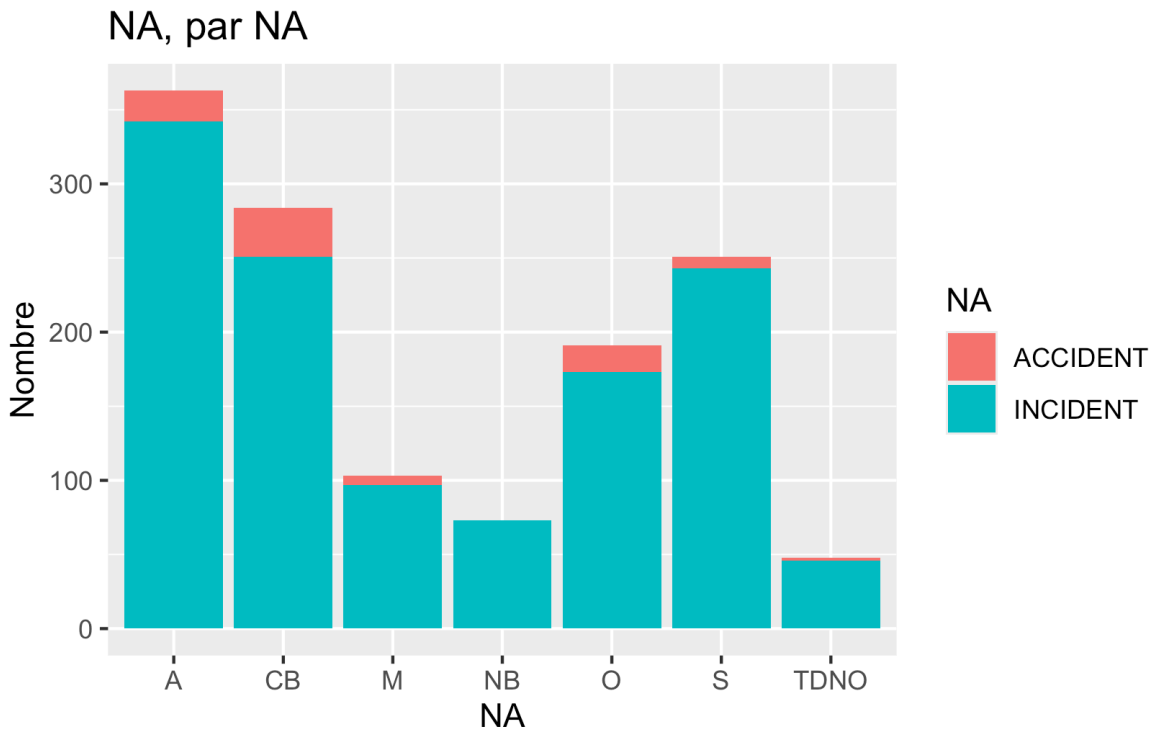
```
initials <- function(str)
  {
  regex <- '(?<=^|\\s|-)[[:alpha:]]'
  ini <- regmatches(str, gregexpr(regex, str, perl=TRUE))
  toupper(sapply(ini, paste0, collapse=''))
  }
```

Distribution des accidents et incidents

```
dist_incidents <- donnees_conv3 %>%
  select(PROVINCE,OCC_TYPE,PRODUCT_TYPE) %>%
  group_by(PROVINCE=factor(initials(PROVINCE)),OCC_TYPE=factor(OCC_TYPE)) %>%
  summarise(N = n())
```

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

```
ggplot(dist_incidents,aes(x=PROVINCE,fill=OCC_TYPE)) +
  geom_bar(aes(y=N),stat="identity") +
  ggtitle(paste0(datadict[datadict$Nom.de.colonne.PODS=="OCC_TYPE"][2],", par ",datadict[data
  scale_fill_discrete(name=datadict[datadict$Nom.de.colonne.PODS=="OCC_TYPE"][2]) +
  xlab(datadict[datadict$Nom.de.colonne.PODS=="PROVINCE"][2]) +
  ylab("Nombre")
```



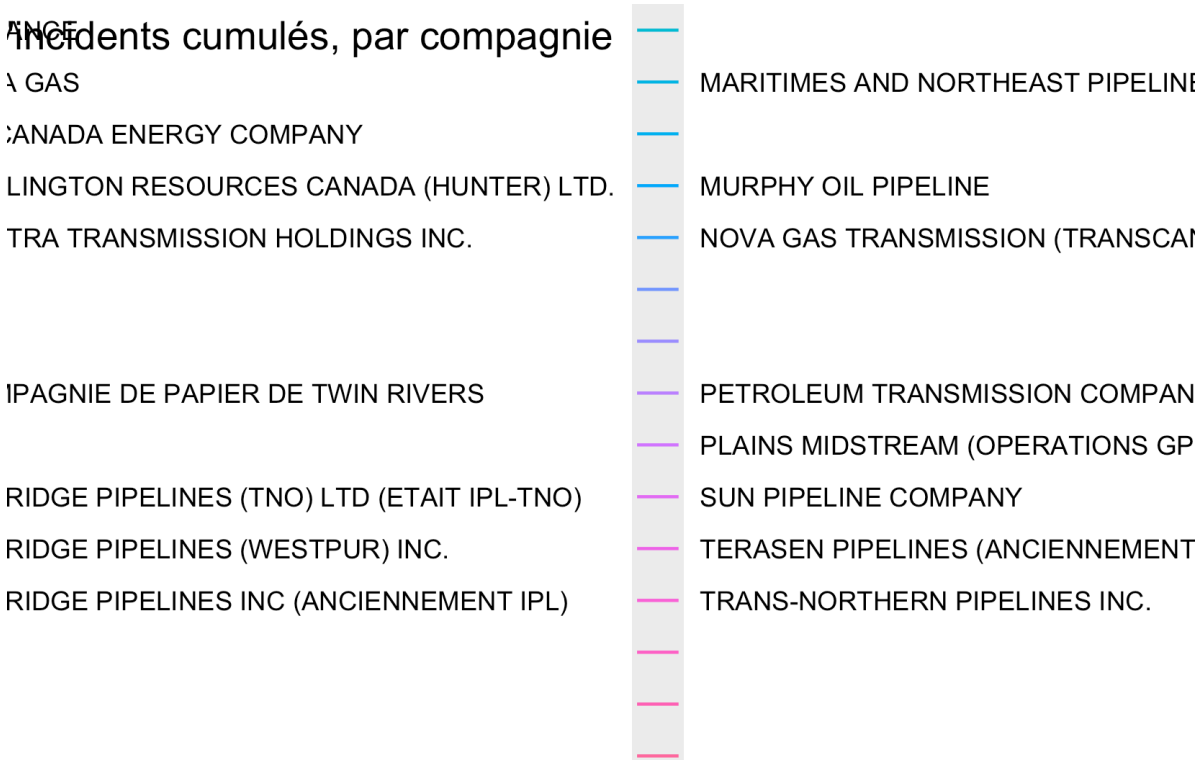
Série chronologique

```
data_serie <- donnees_conv3 %>%
  select(PIPELINE_CO,OCC_DATE,QUANTITY_LOST) %>%
  mutate(N=1, date_clean = parse_date_time(OCC_DATE, orders = "%Y-%m-%d")) %>%
  arrange(PIPELINE_CO,date_clean) %>%
  group_by(PIPELINE_CO,date_clean) %>%
  summarise(Nsum = sum(N)) %>%
  mutate(Ncumsum = cumsum(Nsum))
```

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

Nombre d'évènements cumulés

```
ggplot(data_serie,aes(x=date_clean,y=Ncumsum,color=PIPELINE_CO)) +
  geom_line(stat="identity") +
  xlab("Date") +
  ylab("Nombre d'indicents cumulés") +
  ggtitle("Nombre d'incidents cumulés, par compagnie")
```



Arbre de décision

```
tree1 <- rpart(OCC_TYPE~FACILITY_TYPE+PROD_RELEASED_IND+PIPELINE_TYPE_CO+ENVIRO_DAMAGE_IND+PI
summary(tree1)
```

```
Call:
rpart(formula = OCC_TYPE ~ FACILITY_TYPE + PROD_RELEASED_IND +
    PIPELINE_TYPE_CO + ENVIRO_DAMAGE_IND + PROPERTY_DAMAGE_IND +
    FIRE_IND + EXPLOSION_IND, data = donnees_conv3, method = "class")
  n= 1313

          CP nsplit rel error    xerror       xstd
1 0.38636364      0 1.0000000 1.0000000 0.10296612
2 0.01136364      1 0.6136364 0.6136364 0.08177013
3 0.01000000      3 0.5909091 0.6818182 0.08598766

Variable importance
           FIRE_IND PROPERTY_DAMAGE_IND       FACILITY_TYPE       EXPLOSION_IND
                 89                   6                   3                   2
   PIPELINE_TYPE_CO
                  1

Node number 1: 1313 observations,    complexity param=0.3863636
  predicted class=INCIDENT  expected loss=0.06702209  P(node) =1
    class counts:    88  1225
   probabilities: 0.067 0.933
  left son=2 (88 obs) right son=3 (1225 obs)
  Primary splits:
      FIRE_IND            splits as  RL, improve=73.962500, (0 missing)
      PROPERTY_DAMAGE_IND splits as  RL, improve=11.857030, (0 missing)
      PROD_RELEASED_IND   splits as  LR, improve= 7.269974, (0 missing)
      EXPLOSION_IND       splits as  RL, improve= 5.896740, (0 missing)
      ENVIRO_DAMAGE_IND   splits as  RL, improve= 3.017906, (0 missing)
  Surrogate splits:
      PROPERTY_DAMAGE_IND splits as  RL, agree=0.936, adj=0.045, (0 split)
      EXPLOSION_IND       splits as  RL, agree=0.934, adj=0.011, (0 split)

Node number 2: 88 observations,    complexity param=0.01136364
  predicted class=ACCIDENT  expected loss=0.3068182  P(node) =0.06702209
    class counts:    61    27
   probabilities: 0.693 0.307
  left son=4 (9 obs) right son=5 (79 obs)
  Primary splits:
      PROPERTY_DAMAGE_IND splits as  RL,          improve=1.8875140, (0 missing)
      FACILITY_TYPE       splits as  RLR-R--R-RL, improve=0.9677983, (0 missing)
      PROD_RELEASED_IND   splits as  RL,          improve=0.7032468, (0 missing)
      PIPELINE_TYPE_CO    splits as  -L-RR,       improve=0.4935942, (0 missing)
  Surrogate splits:
```

```
      FACILITY_TYPE splits as  RRR-R--R-RL, agree=0.92, adj=0.222, (0 split)
      EXPLOSION_IND splits as  RL,          agree=0.92, adj=0.222, (0 split)

Node number 3: 1225 observations
  predicted class=INCIDENT  expected loss=0.02204082  P(node) =0.9329779
    class counts:    27  1198
   probabilities: 0.022 0.978

Node number 4: 9 observations
  predicted class=ACCIDENT  expected loss=0  P(node) =0.006854532
    class counts:     9     0
   probabilities: 1.000 0.000

Node number 5: 79 observations,    complexity param=0.01136364
  predicted class=ACCIDENT  expected loss=0.3417722  P(node) =0.06016756
    class counts:    52    27
   probabilities: 0.658 0.342
  left son=10 (65 obs) right son=11 (14 obs)
  Primary splits:
      FACILITY_TYPE     splits as  LLL-L--R-LR, improve=1.7948530, (0 missing)
      PIPELINE_TYPE_CO  splits as  -L-RR,       improve=0.7084829, (0 missing)
      PROD_RELEASED_IND splits as  RL,          improve=0.1797718, (0 missing)
  Surrogate splits:
      PIPELINE_TYPE_CO splits as  -L-RR, agree=0.899, adj=0.429, (0 split)

Node number 10: 65 observations
  predicted class=ACCIDENT  expected loss=0.2923077  P(node) =0.04950495
    class counts:    46    19
   probabilities: 0.708 0.292

Node number 11: 14 observations
  predicted class=INCIDENT  expected loss=0.4285714  P(node) =0.0106626
    class counts:     6     8
   probabilities: 0.429 0.571
```

Affichage de l'arbre de décision

```
rpart.plot::rpart.plot(tree1, main = paste0("Arbre de classification pour ",
                           datadict[datadict$Nom.de.colonne.PODS=="OCC_TYPE"]$Nom.du.champ))
```

Arbre de classification pour Type d'événement