TFM: Análisis predictivo de incidentes navales en EEUU, 2002 - 2015

Anexo 3.4. Preprocesado MergedActivity y VesselBalancedSample

Oscar Antón

diciembre de 2023

Carga de librerías

```r
# Librería                          # Propósito
library(tidyverse)                   # Sintaxis para el manejo de datos. Incluye dplyr, ggplo
t2, etc.
library(data.table)                 # Manejo eficiente de conjuntos de datos
library(leaflet)                    # Representación geográfica
library(tictoc)                     # Benchmarking (tiempo de cómputo)

source("../4.Functions/myCustomFunctions.R")
```

```r
# Guardar datos o no
save_switch <- 0
# Utilizado para no tener que procesar todos los datos en cada renderizado
```

1. Incidentes: MergedActivity

1.1. Carga de dataframes de origen

```r
# Cargar el dataframe de Eventos
Events <- as.data.table(readRDS("DataCasualtyAndPollution/Events.rds"))
# Cargar el dataframe de Barcos
Vessel <- as.data.table(readRDS("DataCasualtyAndPollution/Vessel.rds"))

# Cargar el dataframe de estaciones marítimas
WeatherOcean <- as.data.table(readRDS("DataWeatherOcean/WeatherOcean.rds"))
# Cargar el dataframe de estaciones terrestres
WeatherRiver <- as.data.table(readRDS("DataWeatherRiver/WeatherRiver.rds"))

# Cargar el dataframe de daños personales
Injury <- as.data.table(readRDS("DataCasualtyAndPollution/Injury.rds"))
# Cargar el dataframe de contaminación
VslPoll <- as.data.table(readRDS("DataCasualtyAndPollution/VslPoll.rds"))

# Cargar el dataframe de Actividades
Activity <- as.data.table(readRDS("DataCasualtyAndPollution/Activity.rds"))
```

1.2. Preselección inicial de variables

```r
# Del dataframe de Eventos
Events <- Events %>%
  select(activity_id, vessel_id, vessel_name, vessel_class, waterway_name, event_type, dam
age_status, latitude, longitude, date, hour, region, watertype)

# Del dataframe de Barcos. Solo barcos registrados en Events
Vessel <- Vessel %>%
  select(vessel_id, gross_ton, length, flag_abbr, classification_society, solas_desc, imo_
number, build_year) %>%
  filter(vessel_id %in% Events$vessel_id)

# Del dataframe de estaciones marítimas. Renombrar a minúsculas
WeatherOcean <- WeatherOcean %>%
  select(ACTIVITY_ID, WIND_SPEED, VISIBILITY, AIR_TEMP, WAVE_HGT) %>%
  rename_all(tolower)

# Del dataframe de estaciones terrestres
WeatherRiver <- WeatherRiver %>%
  mutate(WIND_SPEED = AWND) %>%
  mutate(AIR_TEMP = (TMAX + TMIN) / 2) %>%
  select(ACTIVITY_ID, WIND_SPEED, AIR_TEMP) %>%
  rename_all(tolower)

# Del dataframe de daños personales
Injury <- Injury %>%
  select(activity_id, vessel_id, accident_type, casualty_type_desc)

# Del dataframe de contaminación
VslPoll <- VslPoll %>%
  select(activity_id, vessel_id, chris_cd, discharge_amnt_total, damage_status)
```

1.3. Unión de datos

Se tomará como base el dataframe Events

```
# Eventos y datos de barcos
EventsAndVessels <- merge(Events, Vessel, by = "vessel_id", all.x = TRUE) %>%
  distinct(activity_id, vessel_id, event_type, .keep_all = TRUE)

# Meteorología terrestre
EventsRiver <- EventsAndVessels %>%
  filter(watertype == "river")
EventsRiverWeather <- merge(EventsRiver, WeatherRiver, by = "activity_id", all.y = FALSE)
%>%
  unique()
EventsRiverWeather$visibility <- NA
EventsRiverWeather$wave_hgt <- NA

# Meteorología marítima
EventsOcean <- EventsAndVessels %>%
  filter(watertype == "ocean")
EventsOceanWeather <- merge(EventsOcean, WeatherOcean, by = "activity_id", all.x = TRUE) %
>%
  unique()

# Unión vertical River + Ocean. Registros ordenados por fecha y id
MergedActivity <- as.data.table(rbind(EventsOceanWeather, EventsRiverWeather) %>%
    # Orden de variables
    select(activity_id,
         date, hour,
         region, latitude, longitude,
         watertype, event_type, damage_status,
         vessel_id, imo_number, vessel_name, vessel_class,
         build_year, gross_ton, length,
         flag_abbr, classification_society, solas_desc,
         air_temp, wind_speed, wave_hgt, visibility) %>%
    # Orden de observaciones
    arrange(date, activity_id))

# Evaluación de daños
MergedActivity$damage_assessment <- Activity$damage_assessment[match(Events$activity_id, A
ctivity$activity_id)]

# Daños personales
MergedActivity$casualty <- Injury$casualty_type_desc[match(Events$activity_id, Injury$acti
vity_id)]

# Contaminación
MergedActivity$pollution  <-  VslPoll$chris_cd[match(Events$activity_id,  VslPoll$activity_i
d)]
```

1.4. Calidad de datos Merged Activity

```
# Filtrado de las observaciones con 6 o más NAs
MergedActivity <- MergedActivity[rowSums(is.na(MergedActivity)) < 6]
```

```
# Filtrado de observaciones con valores inverosímiles
MergedActivity <- MergedActivity %>%
  filter(gross_ton >= 1 & gross_ton <= 250000) %>%
  filter(build_year >= 1800 & build_year <= 2015 ) %>%
  filter(length >= 1 & length <= 1250) %>%
  distinct(activity_id, vessel_id, event_type, .keep_all = TRUE)
```

1.5. Cálculo de variables para los modelos de clasificación

1.5.1. event_class

La variable event_type, toma hasta 26 valores diferentes. Se va a reducir esta variabilidad agrupando niveles en las siguientes categorías por orden de importancia:

```
1. Critical Events: Sinking, Implosion, Capsize, Loss of Stability, Vessel Maneuverabilit
y, Set Adrift, Abandonment
2. Onboard Emergencies: Loss of Electrical Power, Fire, Emergency Response, Explosion, Flo
oding, Personnel Casualties, Falls into Water
3. Maritime Accidents: Grounding, Allision, Collision
4. Material Issues: Material Failure (Vessels), Material Failure (Non-vessels), Material F
ailure (Diving), Blowout
5. Third-party Damages: Damage to the Environment, Damage to Cargo, Fouling, Evasive Maneu
vers, UNSPECIFIED
```

```
MergedActivity <- MergedActivity %>%
  mutate(event_class = case_when(
     event_type %in% c("Sinking", "Implosion", "Capsize", "Loss of Stability", "Vessel Man
euverability", "Set Adrift", "Abandonment") ~ "Critical Events",
     event_type %in% c("Loss of Electrical Power", "Fire", "Emergency Response", "Explosio
n", "Flooding", "Personnel Casualties", "Falls into Water") ~ "Onboard Emergencies",
     event_type %in% c("Grounding", "Allision", "Collision") ~ "Maritime Accidents",
     event_type %in% c("Material Failure (Vessels)", "Material Failure (Non-vessels)", "Ma
terial Failure (Diving)", "Blowout") ~ "Material Issues",
     event_type %in% c("Damage to the Environment", "Damage to Cargo", "Fouling", "Evasive
Maneuvers", "UNSPECIFIED") ~ "Third-party Damages"
     )
  )
```

1.5.2. Antiguedad

Al conocer la fecha de botadura y la de incidente, se puede calcular la antiguedad en ese momento para aportar información más precisa

```
MergedActivity <- MergedActivity %>%
 mutate(age = year(as.Date(date)) - year(as.Date(paste0(build_year, "-01-01")))) %>%
 relocate(age, .after = "build_year")
```

```
str(MergedActivity)
```

```
## Classes 'data.table' and 'data.frame':   68000 obs. of  28 variables:
##  $ activity_id           : int  1475897 1475897 1477008 1477373 1477402 1484262 1485352
1485352 1598058 1475186 ...
##  $ date                  : Date, format: "2002-01-01" "2002-01-01" ...
##  $ hour                  : chr  "03:45" "03:45" "13:53" "18:10" ...
##  $ region                : chr  "Mississippi" "Mississippi" "East Coast" "Mississippi"
...
##  $ latitude              : num  37 37 39.3 31.5 30.6 ...
##  $ longitude             : num  -88.3 -88.3 -76.4 -88 -88 ...
##  $ watertype             : chr  "river" "river" "ocean" "river" ...
##  $ event_type            : chr  "Grounding" "Grounding" "Damage to the Environment" "Ma
terial Failure (Vessels)" ...
##  $ damage_status         : chr  "Undamaged" "Undamaged" "Undamaged" "Damaged" ...
##  $ vessel_id             : int  193888 219821 125903 208043 441276 464890 198843 198843
226659 84253 ...
##  $ imo_number            : chr  "" "" "" "" ...
##  $ vessel_name           : chr  "ING 4508" "JOHN M. RIVERS" "MISS DIANE" "PASCAGOULA"
...
##  $ vessel_class          : chr  "Barge" "Towing Vessel" "Fishing Vessel" "Towing Vesse
l" ...
##  $ build_year            : chr  "1981" "1981" "1973" "1982" ...
##  $ age                   : int  21 21 29 20 6 5 21 21 25 39 ...
##  $ gross_ton             : int  1065 932 19 227 764 823 888 888 38412 13 ...
##  $ length                : num  200 135.6 38.2 78.8 200 ...
##  $ flag_abbr             : chr  "US" "US" "US" "US" ...
##  $ classification_society: chr  "UNSPECIFIED" "UNSPECIFIED" "UNSPECIFIED" "UNSPECIFIED"
...
##  $ solas_desc            : chr  "Non SOLAS" "Non SOLAS" "Non SOLAS" "Non SOLAS" ...
##  $ air_temp              : num  -38.5 -38.5 -17.1 11 41.2 ...
##  $ wind_speed            : num  NA NA 66.2 NA 85.7 ...
##  $ wave_hgt              : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ visibility            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ damage_assessment     : int  100 100 5600 5600 1000 95000 95000 10000 10000 40000
...
##  $ casualty              : chr  NA NA NA NA ...
##  $ pollution             : chr  NA NA NA NA ...
##  $ event_class           : chr  "Maritime Accidents" "Maritime Accidents" "Third-party
Damages" "Material Issues" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

1.6. Guardado de datos MergedActivity

```
if (save_switch  == 1) {
 # Llamado a la función para guardar(dataframe, nombre de carpeta de destino)
 loggedsave(MergedActivity, "DataMergedActivity")
}
```

---

2. Barcos con y sin incidentes: VesselBalancedSample

```
# Cargar el dataframe de todos los barcos
Vessel <- as.data.table(readRDS("DataCasualtyAndPollution/Vessel.rds"))
```

2.1. Acotación de observaciones para valores verosímiles

```
# Se aplican varios filtros
VesselNoActivity <- Vessel %>%
  filter(!vessel_id %in% MergedActivity$vessel_id) %>%
  filter(gross_ton >= 1 & gross_ton <= 250000) %>%
  filter(build_year >= 1800 & build_year <= 2015 ) %>%
  filter(length > 1 & length <= 1250) %>%
  unique()
```

2.2. Guardado de datos de barcos sin incidentes

```
if (save_switch  == 1) {
# Llamado a la función para guardar(dataframe, nombre de carpeta de destino)
loggedsave(VesselNoActivity, "DataMergedActivity")
}
```

2.3. Selección de barcos sin incidentes

```
# Barcos con incidentes
VesselActitivity <- MergedActivity %>%
    select(vessel_id, imo_number, vessel_name, vessel_class, build_year,
         gross_ton, length,
         flag_abbr, classification_society, solas_desc,
         event_type, damage_status) %>%
  unique()
```

```r
# Selección al azar
# Se van a seleccionar tantos registros como en el dataframe MergedActivity
VesselNoActivitySample <- VesselNoActivity %>%
  select(vessel_id, imo_number, vessel_name, vessel_class, build_year,
         gross_ton, length,
         flag_abbr, classification_society, solas_desc) %>%
  sample_n(nrow(VesselActitivity), replace = FALSE) %>%
  mutate(event_type = "No event", damage_status = "Undamaged")
```

2.4. Fusión de datos de barco con y sin incidentes

```r
VesselBalancedSample <- rbind(VesselNoActivitySample, VesselActitivity) %>%
  unique()
```

```r
if (save_switch  == 1) {
# Llamado a la función para guardar(dataframe, nombre de carpeta de destino)
loggedsave(VesselBalancedSample, "DataMergedActivity")
}
```