Densidad Kernel Rasters

Eduardo Alanís

2025-03-12

¿Qué es la Densidad Kernel?

La herramienta Densidad kernel calcula la densidad de las entidades en la vecindad de esas entidades. Puede calcularse para las entidades de punto y de línea.

Esta técnica ayuda a entender patrones en la distribución de datos. Por ejemplo, se puede usar para ver qué zonas tienen más crímenes o cómo las carreteras afectan el entorno natural. También permite asignar diferente importancia a ciertos puntos, como un edificio con varias viviendas o ciertos delitos más graves que otros.

Que realiza el código

Se utilizaron los datos de DENUE descargados de INEGI y se filtró la información para el estado de Hidalgo. Para el análisis se emplearon las siguientes columnas:

- codigo_act
- nombre_act
- per_ocu
- cve_mun
- cve_loc
- latitud
- longitud

Adicionalmente, se usó el documento "Clasificación ramas de DENUE", que actúa como diccionario al asignar un grupo a cada actividad basado en el codigo_act. Con esta clasificación, se calculó la densidad Kernel, generando un raster para cada grupo. Cada raster destaca las ubicaciones de las actividades de cada grupo según un aproximado del número de trabajadores.

Código

Carga las librerías necesarias.

```
library(sf)
library(readxl)
library(raster)
library(spatstat)
```

Primero, define tu directorio de trabajo. Luego, la variable archivos buscará todos los archivos dentro de ese directorio, específicamente los del DENUE, que previamente descargaste y descomprimiste. Después, archivos_csv filtrará solo los archivos con extensión .csv, excluyendo aquellos que contienen la palabra "diccionario", para quedarnos únicamente con las bases de datos del DENUE. Finalmente, la variable nombre_guardado almacenará el semestre correspondiente a la base de datos, asegurando que los archivos guardados sigan un orden.

```
# setwd("C:/Users/eagel/OneDrive/Escritorio/Lalo/Escuela/Practicas/Chamba/")
archivos = list.files("C:/Users/eagel/OneDrive/Escritorio/Lalo/Escuela/Practicas/Chamba/Marzo/Archivos
```

```
archivos_csv = grep("\\.csv$", archivos, value = TRUE)
archivos_csv = subset(archivos_csv, !grepl("diccionario", archivos_csv))

nombre_guardado = basename(dirname(dirname(archivos_csv)))
nombre_guardado = sub(pattern = "denue_13_",replacement = "", x = nombre_guardado)
nombre_guardado = sub(pattern = "_csv",replacement = "", x = nombre_guardado)
```

Dado que el código utiliza un for para repetir este proceso automáticamente, ahora explicaremos qué ocurre en una de esas vueltas del ciclo.

En esta parte se define la variable denue la cual abre el archivo csv i para posteriormente fitrar esta base apartir de las columnas latitud y longititud eliminando todas aquellas las que tienen en estos apartados 0, ya que esto indica que no existe una punto de geolocalización para ellos.

Posteriormente se define la variable **ruta** en la cual indicaras en la direccion en la que vas a querer guardar todos los archivos que se generaran. Donde al definir una ruta esta creara una carpeta con el nombre del guardado con anterioridad que indica el Año y semestre del DENUE.

Se define la variable denue, que abre el archivo CSV y filtra la base de datos eliminando las filas donde la latitud y longitud sean 0, ya que esto indica la ausencia de geolocalización.

Luego, se crea la variable ruta, donde se especifica la carpeta en la que se guardarán los archivos generados. En direccion se creara una carpeta automáticamente con el nombre correspondiente al año y semestre del DENUE.

```
i = 2
denue = read.csv(archivos_csv[i])
    denue = denue |>
        dplyr::filter(latitud != 0, longitud != 0)

ruta = "C:/Users/eagel/OneDrive/Escritorio/Lalo/Escuela/Practicas/Chamba/Marzo/Cosas Realizadas/Rasters
nombre_carpeta = nombre_guardado[i]

# Crear la ruta destino con el nombre del archivo original
destino_guardado = file.path(ruta, nombre_carpeta)

# Crear la carpeta destino si no existe
if (!dir.exists(destino_guardado)) {
    dir.create(destino_guardado, recursive = TRUE)
}
```

En este paso, se seleccionan solo las columnas de interés de la base del DENUE: codigo_act, nombre_act, per_ocu, cve_mun, cve_loc, latitud y longitud.

Luego, se crea una nueva columna llamada rama, que contiene los primeros cuatro dígitos de codigo act.

De forma similar, se agrega la columna per_ocu_media, que convierte la columna per_ocu en valores numéricos usando el punto medio del intervalo proporcionado. Para esto, se utiliza la función media_trabajadores.

```
"31 " = 40.5,
    "51 " = 75.5,
    "101" = 175.5,
    "251" = 251

))
}
denue$per_ocu_media = sapply(substr(denue$per_ocu, start = 1, stop = 3),media_trabajadores,simplify =
```

Se define la variable classificador_hoja, que contiene el archivo de Excel con la clasificación de Ramas del DENUE. Luego, este archivo se une ordenadamente con la base del DENUE en la variable denue_prueba_shp.

```
classificador_hoja = readxl::read_excel("C:/Users/eagel/OneDrive/Escritorio/Lalo/Escuela/Practicas/Cham'
colnames(classificador_hoja)[3] = "grupo"
classificador_hoja$grupo = gsub("[0-9]+", "", classificador_hoja$grupo)
denue_prueba_shp = merge(x = denue, y = classificador_hoja, by.x = 'rama', by.y = 'Rama', all.x = T )
unique(denue_prueba_shp$rama[denue_prueba_shp$grupo|>is.na()])
```

character(0)

En esta parte, se crea la variable shapefile, que transforma denue_prueba_shp en un archivo de tipo sf, donde la geometría se basa en la latitud y longitud, utilizando la proyección CRS 4326.

De manera similar, se crea la variable base, que carga un archivo donde el estado de Hidalgo esta cubierto por 0's y todo lo demas NA. A partir de este, se calculan sus dimensiones y se almacenan en la variable window.

Guardamos en la variable grupo todos los distintos grupos y vamos ha usar a variable municipios por los nombre de los municipios.

Se crea la variable shapefile_grupo filtrando el shapefile original para quedarse solo con los datos del grupo actual.

La variable coords se obtiene transformando el sistema de coordenadas de shapefile_grupo para que coincida con el de la variable base y luego se extraen las coordenadas de los puntos.

Se construye la variable puntos_ppp de la siguiente forma:

- Se usa inside.owin para seleccionar solo las coordenadas que están dentro del área de interés, definida previamente como window.
- La función ppp convierte estos puntos en un objeto ppp (un conjunto de puntos en Hidalgo).

La variable pesos se forma a partir de la columna per_ocu_media del shapefile_grupo, seleccionando únicamente los valores correspondientes a los puntos que caen dentro del área window.

La variable dens calcula la densidad Kernel y la convierte en un raster. A este raster se le asigna el sistema de coordenadas (crs) y la extensión de Hidalgo, de modo que el área de Hidalgo se marca con 1 y el resto con NA.

Finalmente guarda o exporta este kernel.

```
for(grupo_I in grupos){
    shapefile_grupo = shapefile |> dplyr::filter(grupo == grupo_I)
    coords = st_coordinates(shapefile_grupo|>st_transform(crs(base)))  # Extraer las coordenadas de l
```

```
puntos_ppp = ppp(x = coords[inside.owin(x = coords[, 1], y = coords[, 2], window), 1], y = coords[
pesos = shapefile_grupo$per_ocu_media[inside.owin(x = coords[, 1], y = coords[, 2], window)]
dens = density(puntos_ppp,sigma=c(2000,2000),dimyx=c(998,978),weights =1e6*100*unlist(pesos, use.naccrs(dens) = crs(base)
extent(dens) = extent(base)
dens = dens*(base+1)
writeRaster(dens,file.path(destino_guardado, paste0(grupo_I,".tiff")), overwrite = T)
}
```