Soraya Sakka Afyouni – u22125158

Spark con R-Studio

En primer lugar, se cargan las librerías y la ruta necesaria para proceder a hacer un estudio del fichero, así como la conexión a Spark. Para ello, se ha hecho uso del siguiente código:

```
pacman::p_load(httr, tidyverse,leaflet,janitor,readr,sparklyr)

url<-
"https://sedeaplicaciones.minetur.gob.es/ServiciosRESTCarburantes/PreciosCarburantes/EstacionesTerrestres/"

httr::GET(url)

library(dplyr)

library(sparklyr)

sc<-spark_connect(master="local")
```

En esta ocasión, se está usando la versión 3.0.0 de Spark.

Ejercicio a:

Para una correcta manipulación de los datos, es necesario primero hacer una limpieza del dataset, picando el siguiente código:

```
jsonlite::fromJSON(url)

ds<- jsonlite::fromJSON(url)

ds<-ds$ListaEESSPrecio

ds<- ds %>%  as_tibble() %>% clean_names()

ds<-ds%>% type_convert(locale = locale(decimal_mark = ","))  %>% view() %>% clean_names()
```

Con esto, conseguimos crear la variable “ds”, la cual será la tabla con la que iremos trabajando y haciendo modificaciones sobre ella y cambiar ciertas anomalías que vienen por defecto, como es el nombre de las distintas variables de la tabla, los formatos (texto, decimal, etc.), entre otros.

En este caso, consideramos como limpieza y anomalía, también, las comas, puesto que, en caso de dejarlo así, no se puede poner en formato “decimal” como debería ser.

Hecho esto, queda la tabla con un formato como el siguiente:

Filter

	c_p	direccion	horario
1	02250	AVENIDA CASTILLA LA MANCHA, 26	L-D: 07:00-22:00
2	02152	CR CM-332, 46,4	L-D: 7:00-23:00
3	02001	CALLE PRINCIPE DE ASTURIAS (POLÍGONO DE ROMI...	L-D: 06:00-21:30
4	02001	CALLE FEDERICO GARCIA LORCA, 5	L-S: 05:00-23:00
5	02001	CALLE FEDERICO GARCIA LORCA, 1	L-D: 24H
6	02001	AVENIDA 1º DE MAYO, S/N	L-S: 08:00-22:00; D: 09:00-21:00
7	02005	CL PASEO DE LA CUBA, 15	L-D: 06:00-22:00
8	02005	PASEO CUBA (LA), 36	L-D: 06:00-23:00
9	02005	AVENIDA MENÉNDEZ PIDAL, 58	L-D: 24H
10	02005	AVENIDA ESCRITOR RODRIGO RUBIO, 3	L-S: 09:00-21:30
11	02002	CALLE HERMANOS FALCÓ, 2	L-D: 06:30-22:30

A continuación, nos interesa crear una columna nueva llamada “Low-cost” en la que distinguiremos las columnas “low-cost” de las que no lo son, así como el cálculo de la media a nivel ccaa y provincias. Para ello:

Este primer código usando la función “mutate” para crear la nueva columna, en la que incluimos las principales gasolineras. El resultado que obtendremos será de “true” y “false”:

```
ds_lowcost<- ds %>%mutate(low_cost=rotulo%in%c('REPSOL','CAMPSA','BP','SHELL','GALP','CEPSA')) %>%
view()
```

Este segundo código para reemplazar “true” y “false” por “no-lowcost” y “lowcost”:

```
ds_lowcost$low_cost[ds_lowcost$low_cost==TRUE]<-"No-lowcost"

ds_lowcost$low_cost[ds_lowcost$low_cost==FALSE]<-"Lowcost"
```

Observamos la columna añadida y su separación entre las dos variables en la siguiente imagen adjuntada:

ideess	id_municipio	id_provincia	idccaa	low_cost
4375	52	02	07	Lowcost
5122	53	02	07	No-lowcost
12054	54	02	07	Lowcost
4438	54	02	07	No-lowcost
13933	54	02	07	Lowcost
10765	54	02	07	Lowcost
5195	54	02	07	No-lowcost
14123	54	02	07	No-lowcost
4369	54	02	07	Lowcost
15000	54	02	07	Lowcost
5313	54	02	07	No-lowcost

Los siguientes códigos, para calcular ambas medias, a nivel ccaa y provincias. En estos, usamos la tabla creada en el apartado anterior ya que es en la que se encuentra la distinción entre lowcost y no. Se hace uso de la función “select” para coger todos los tipos de gasolinas y

columnas que nos interesan, hacemos un “group_by” por ID de ccaa o provincias y un “summarise” para calcular las medias. Finalmente, un “view” para ver el resultado. Quedando:

```
media1<- ds_lowcost %>% select(precio_bioetanol, precio_biodiesel, precio_gas_natural_comprimido,precio_gas_natural_licuado, precio_gases_licuados_del_petroleo, precio_gasoleo_a, precio_gasoleo_b, precio_gasoleo_premium, precio_gasolina_95_e10, precio_gasolina_95_e5, precio_gasolina_95_e5_premium, precio_gasolina_98_e10, precio_gasolina_98_e5, precio_hidrogeno, rotulo, idccaa, provincia) %>%

 group_by(idccaa) %>% summarise(media_precio_bioetanol=mean(precio_bioetanol, na.rm=TRUE), mean(precio_biodiesel, na.rm=TRUE), mean(precio_gas_natural_comprimido, na.rm=TRUE), mean(precio_gas_natural_licuado, na.rm=TRUE), mean(precio_gases_licuados_del_petroleo, na.rm=TRUE), mean(precio_gasoleo_a, na.rm=TRUE), mean(precio_gasoleo_b, na.rm=TRUE), mean(precio_gasoleo_premium, na.rm=TRUE), mean(precio_gasolina_95_e5, na.rm=TRUE), mean(precio_gasolina_95_e5_premium, na.rm=TRUE), mean(precio_gasolina_98_e10, na.rm=TRUE), mean(precio_gasolina_98_e5, na.rm=TRUE), mean(precio_hidrogeno, na.rm=TRUE)) %>% view()


media2<- ds_lowcost %>% select(precio_bioetanol, precio_biodiesel, precio_gas_natural_comprimido,precio_gas_natural_licuado, precio_gases_licuados_del_petroleo, precio_gasoleo_a, precio_gasoleo_b, precio_gasoleo_premium, precio_gasolina_95_e10, precio_gasolina_95_e5, precio_gasolina_95_e5_premium, precio_gasolina_98_e10, precio_gasolina_98_e5, precio_hidrogeno, rotulo, idccaa, provincia) %>%

 group_by(provincia) %>% summarise(media_precio_bioetanol=mean(precio_bioetanol, na.rm=TRUE), mean(precio_biodiesel, na.rm=TRUE), mean(precio_gas_natural_comprimido, na.rm=TRUE), mean(precio_gas_natural_licuado, na.rm=TRUE), mean(precio_gases_licuados_del_petroleo, na.rm=TRUE), mean(precio_gasoleo_a, na.rm=TRUE), mean(precio_gasoleo_b, na.rm=TRUE), mean(precio_gasoleo_premium, na.rm=TRUE), mean(precio_gasolina_95_e5, na.rm=TRUE), mean(precio_gasolina_95_e5_premium, na.rm=TRUE), mean(precio_gasolina_98_e10, na.rm=TRUE), mean(precio_gasolina_98_e5, na.rm=TRUE), mean(precio_hidrogeno, na.rm=TRUE)) %>% view()
```

Imagen de ejemplo de la media calculada a nivel ccaa para cada tipo de gasolina:

idccaa	media_precio_bioetanol	mean(precio_biodiesel, na.rm = TRUE)	mean(precio_gas_natural_comprimido, na.rm = TRUE)
01	NaN	1.308444	1.98300
02	NaN	NaN	1.47975
03	1.729	NaN	1.46666
04	NaN	NaN	Na
05	NaN	NaN	0.92000
06	NaN	1.301000	Na
07	NaN	NaN	1.93309
08	NaN	1.389000	1.50833
09	1.672	1.281308	1.92509
10	NaN	1.293500	2.13111
11	NaN	1.348000	1.51300

Para continuar, imprimimos mapa de las gasolineras 10 más caras y 20 más baratas. Para evitar sacar un mapa para cada tipo de gasolina, he escogido como ejemplo el precio_biodiesel. Siendo los códigos:

```
ds %>% select(rotulo, latitud, longitud_wgs84, precio_biodiesel, localidad, direccion) %>% top_n(10, precio_biodiesel) %>%  leaflet() %>% addTiles() %>%  addCircleMarkers(lng = ~longitud_wgs84, lat = ~latitud, popup = ~rotulo,label = ~precio_biodiesel)
```

```
ds %>% select(rotulo, latitud, longitud_wgs84, precio_biodiesel, localidad, direccion) %>% top_n(-20, precio_biodiesel) %>%  leaflet() %>% addTiles() %>%  addCircleMarkers(lng = ~longitud_wgs84, lat = ~latitud, popup = ~rotulo,label = ~precio_biodiesel)
```

Siendo los mapas (la primera el topo 10 y la segunda el top 20):

Ejercicio b:

Para sabes cuántas gasolineras hay en Madrid y Cataluña usamos los siguientes códigos, en los que los resultados son:

Cataluña:

	low_cost	n
1	Lowcost	827
2	No-lowcost	653

Madrid:

	low_cost	n
1	Lowcost	319
2	No-lowcost	473

Para hallar esos resultados, se ha hecho uso de los códigos siguientes, en los que se hace un "select" incluyendo las columnas que nos interesan para este aspecto, un "filtro" para poner cuántas ccaa son para cada comunidad y un "count" para que salga en un solo nº cuántas lowcost y no lowcost hay en cada una.

```
comunidad_madrid<-ds_lowcost %>% select(idccaa, low_cost, provincia) %>% filter(idccaa=="13") %>% count(low_cost)
```

```
comunidad_cataluna<-ds_lowcost %>% select(idccaa, low_cost, provincia) %>% filter(idccaa=="09") %>% count(low_cost)
```

Para calcular la media, precio más bajo y alto para cada comunidad (Madrid y Cataluña), usamos el código siguiente. Select para seleccionar las columnas, filter para hacer un filtro de los que necesitamos, drop_na para eliminar los NAs puesto que en este caso no es objeto de estudio, summarise para calcular max, min y mean de los tipos de gasolinas que nos interesan para este apartado:

```
madrid<-ds_lowcost %>% select(idccaa, low_cost, provincia, precio_gasoleo_a, precio_gasolina_95_e5_premium) %>% filter(idccaa=="13") %>%  drop_na() %>%
```

```
 summarise(max(precio_gasoleo_a), min(precio_gasoleo_a), mean(precio_gasoleo_a), max(precio_gasolina_95_e5_premium), min(precio_gasolina_95_e5_premium), mean(precio_gasolina_95_e5_premium))
```

"De igual manera aplicado a Cataluña"

Ejemplo de esto es la siguiente imagen en la que observamos max, min y mean de precio_gasoleo_a para la Comunidad de Madrid:

max(precio_gasoleo_a)	min(precio_gasoleo_a)	mean(precio_gasoleo_a)
1.465	1.345	1.431784

Ejercicio C:

Aquí, nuestro objetivo es conocer a nivel municipio cuántas gasolineras son lowcost, cuántas no, precio medio, max y min de toda España excepto las 4 grandes ciudades. El código picado es el siguiente, en el que con un select se seleccionan las columnas necesarias para este apartado, con un group_by seleccionamos municipio y lowcost ya que es lo que nos interesa, filtramos por municipio quitando las 4 big cities, y con un summarise calculamos media, max y min.

```
no_big_cities <- ds_lowcost %>% select(idccaa, low_cost, precio_gasoleo_a, precio_gasolina_95_e5_premium, municipio, id_municipio) %>% group_by(municipio, low_cost) %>%

 filter(!municipio %in% c("Madrid", "Barcelona", "Sevilla", "Valencia")) %>% summarise(mean(precio_gasoleo_a, na.rm = TRUE), mean(precio_gasolina_95_e5_premium, na.rm = TRUE), max(precio_gasoleo_a, na.rm = TRUE), max(precio_gasolina_95_e5_premium, na.rm = TRUE), min(precio_gasoleo_a, na.rm = TRUE), min(precio_gasolina_95_e5_premium, na.rm = TRUE))
```

Ejercicio D:

Para calcular las gasolineras que están abiertas de Lunes a Viernes las 24h, haciendo así uso de la función “filter” para quedarnos con aquellas que únicamente están disponibles 24/7:

```
no_24_horas<-ds %>% select(c_p,direccion,localidad,municipio,id_provincia,ideess) %>% filter(ds$horario=="L-D: 24H")
```

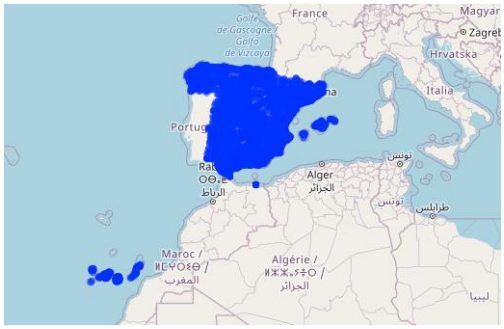
Ejercicio E:

A partir de las direcciones, calculamos cuánta competencia hay en un radio de 1km, 2km y 4km. Para su correcta visualización, adjunto código (para ello, hacemos uso de la biblioteca “leaflet” con la que podemos hacer mapas interactivos, así como “addTiles” y “addCircles”) e imagen del mapa del caso de radio 1km (para los demás, se hace de igual manera):

```
ds %>%  select(rotulo, latitud, longitud_wgs84, precio_gasolina_98_e5, localidad, direccion, ideess) %>%

 leaflet() %>% addTiles() %>%

 addCircles(lng = ~longitud_wgs84, lat = ~latitud, popup = ~rotulo,label = ~ideess, radius = 1)
```



##Hecha la memoria descriptiva, subo al Canvas los diferentes archivos pedidos, el R Script para la correcta visualización y ejecución de los códigos, captura de R Studio para confirmar la conexión a Spark, se ha hecho propietario al profesor en GitHub y a continuación adjunto enlace a GitHub en el que constan los archivos subidos al repositorio:

https://github.com/SorayaSakka/u22125158.Spark