



MÁSTER EN BUSINESS ANALYTICS

MÓDULO II: POSICIONAMIENTO EMPRESARIAL DEL BIG DATA

JOHANA MARISOL GARCÍA LÓPEZ (u22138972)

Caso de uso de en BUSINESS ANALYTICS.

Basados en lo aprendido hasta la fecha y en las prácticas realizadas en clase deberá responder a las preguntas que cumplan los siguientes requerimientos: Un empresario del sector de la hostelería pretende incursionar en el sector de la energía, concretamente en la instalación de gasolineras low-cost, para ello el requiere tomar decisiones de tipo empresarial, que deberá responder usted como en la analista de negocio, este empresario tiene 3 millones de euros y desea invertir en la instalación de gasolineras, pero no quiere pagar franquicias ya que las mismas están sujetas a muchas restricciones, pagos permanente de royalties, no tienen independencia en la toma de decisiones , por lo que se pide:

- A. **De fuentes veraces, lea los archivos que se indican en el anexo, como podrá apreciar, el/los archivos contienen miles de filas por decenas de columnas; solo es posible tratarlos utilizando Spark si queremos respuestas en tiempo real;**

Primero lo que hago es instalar los paquetes necesarios como: httr, tidyverse, leaflet, janitor, readr, sparklyr y luego iré añadiendo los que sean necesarios. Tidyverse es el paquete para data analytics, en janitor lo que hago es apoyarme para limpiar el archivo, con leaflet me apoyare para la creación de los mapas.

El siguiente paso es saber dónde quiere llegar. Voy a coger los datos de GEOPORTAL y me voy a la información actualizada de precios. Elijo el primer URL, el de estaciones de servicios y eso lo almaceno en una variable para luego coger esa URL.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

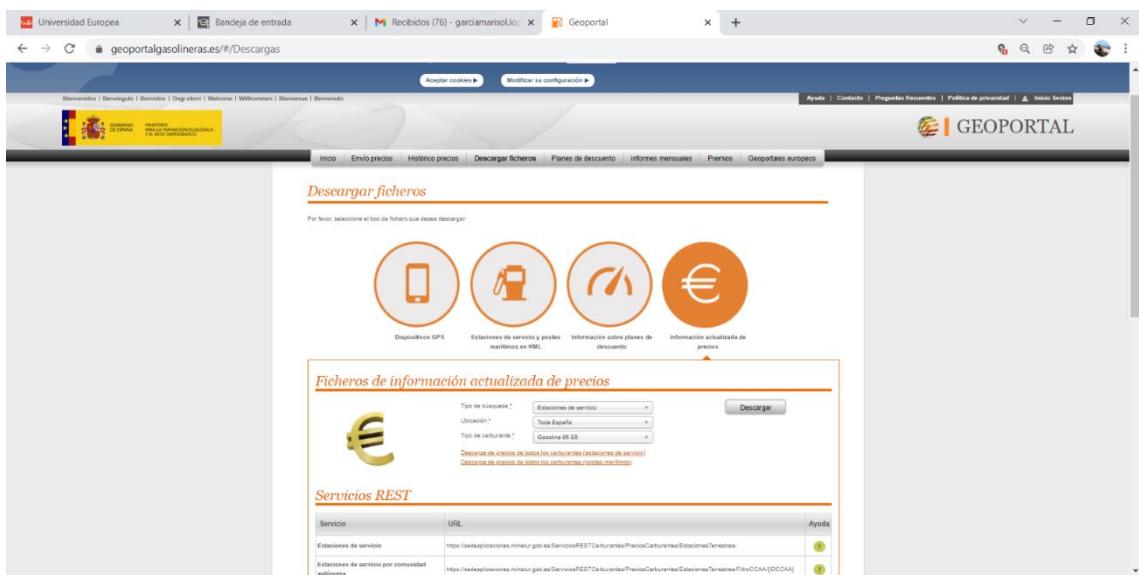


Imagen 1. Cojo los datos de GEOPORTAL

```

#CARGO LIBRERIAS
pacman::p_load(httr, tidyverse, leaflet, janitor, readr, sparklyr)

url <- "https://sedeaplicaciones.minetur.gob.es/ServiciosRESTCarburantes/PreciosCarburantes/EstacionesTerrestres/"
httr::GET(url)

```

```

[R 4.1.2 · ~/]
[1] > pacman::p_load(httr, tidyverse, leaflet, janitor, readr, sparklyr)
[2] > url <- "https://sedeaplicaciones.minetur.gob.es/ServiciosRESTCarburantes/PreciosCarburantes/EstacionesTerrestres/"
[3] > httr::GET(url)
[4] Response [https://sedeaplicaciones.minetur.gob.es/ServiciosRESTCarburantes/PreciosCarburantes/EstacionesTerrestres/]
[5]   Date: 2022-01-03 09:38
[6]   Status: 200
[7]   Content-Type: application/xml; charset=utf-8
[8]   Size: 15.6 MB
[9]   <BINARY BODY>
[10]  >

```

Imagen 2. Cargo librería, cojo datos y compruebo status.

Todos los archivos
están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

Comprobamos en consola status:200 podemos ver que esta perfecto según lo aprendido en clase.

Una vez conseguido los datos lo que hago es con glimpse ver mis datos, presentar mis datos para corregir lo que sea necesario que corresponde al siguiente apartado.

NOTA IMPORTANTE: En mi caso desde clase me da un error que no me deja realizar la conexión con Spark (error del core) que no llego a solucionar, adjunto pantallazo. Intento con el código o directamente desde conexión y no me deja me da error.

```
#CARGO LIBRERIAS
pacman::p_load(httr, tidyverse, leaflet, janitor, readr, sparklyr, xlsx, XML)
url <- "https://sedeaplicaciones.minetur.gob.es/ServiciosRESTCarburantes/PreciosCarburantes/EstacionesTerrestres"
httr::GET(url)
sc <- spark_connect(master = "local") #En mi caso me da error (desde el primer dia) por el core#
#Ejercicio A.i. Limpieza de datos del Dataset
ds <- jsonlite::fromJSON(url)
ds <- ds$listasESSPrecio
ds <- ds %>% as_tibble() %>% clean_names()
View(ds)
ds <- ds %>% type_convert(locale = locale(decimal_mark = ",")) %>% view() %>% clean_names()
#Ejercicio A.ii. Generar informe y explique si encuentra alguna anomalia del punto anterior.
#Este apartado se encuentra detallado en la memoria descriptiva.
#Ejercicio A.iii. Nueva columna llamada low-cost clasificada por low-cost y no low-cost. Y precio promedio
#de todos los combustible
#(Top Level):
```

R 4.1.2 - C:/Users/jmari/OneDrive/Escritorio/spark/

Content-Type: application/xml; charset=utf-8
Size: 15.8 MB
<BINARY BODY>

```
> sc <- spark_connect(master = "local") #En mi caso me da error (desde el primer dia) por el core#
* Using Spark: 3.0.0
Error in system2(file.path(spark_home, "bin", "spark-submit"), "--version", :
  'C:/Users/jmari/AppData\Local/spark/spark-3.0.0-bin-hadoop3.2/bin/spark-submit' not found
> |
```

Imagen 3. Error importante que me da al intentar spark_connect

i. **Limpie el/los dataset(s) (la información debe estar correctamente formateada, por ej. lo que es de tipo texto no debe tener otro tipo que no sea texto), ponga el formato correcto en los números, etc., etc.**

Como no quiero df lo que quiero es tibble, pido un tibble. Como mencione al principio me apoyo en janitor para limpiar mis datos. Lo primero que hago es pedir tibble, luego limpio los nombres todo lo que sea número en formato número y lo que sea carácter en formato carácter.

Todos los archivos
están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

```

1 #CARGO LIBRERIAS
2 pacman::p_load(httr, tidyverse, leaflet, janitor, readr, sparklyr, xlsx, XML)
3
4
5 url <- "https://sedeaplicaciones.minetur.gob.es/ServiciosRESTCarburantes/PreciosCarburantes/EstacionesTerrestres"
6 httr::GET(url)
7
8 sc <- spark_connect(master = "local") #En mi caso me da error (desde el primer dia) por el core
9
10 #Ejercicio A.1. Limpieza de datos del Dataset
11 ds <- jsonlite::fromJSON(url)
12 ds <- janitor::read_tsv(ds)
13 ds <- ds %>% as_tibble() %>% clean_names()
14 View(ds)
15 ds <- ds %>% type_convert(locale = locale(decimal_mark = ",")) %>% view() %>% clean_names()
16
17 #Ejercicio A.ii. Generar informe y explique si encuentra alguna anomalía del punto anterior.
18 #Este apartado se encuentra detallado en la memoria descriptiva.
19
20
21 #Ejercicio A.iii. Nueva columna llamada low-cost clasificada por low-cost y no low-cost. Y precio promedio
22 #de todos los combustibles
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

R Script

Console Terminal Jobs

R 4.1.2 - C:\users\jmarl\OneDrive\Escritorio\spark/

Content-Type: application/xml; charset=utf-8
Size: 15.8 MB
<BINARY BODY>

```

> sc <- spark_connect(master = "local") #En mi caso me da error (desde el primer dia) por el core
Error: file.path(spark_home, "bin", "spark-submit") ==> version
'C:\Users\jmarl\AppData\Local\spark/spark-3.0.0-bin-hadoop3.2/bin/spark-submit' not found

```

Imagen 4. Limpieza de datos

c_p	direccion	horario	latitud	localidad	longitud_wgs84	margen	municipio
1	AVENIDA CASTILLA LA MANCHA, 26	L-D: 07:00-22:00	39.000886	ALBACETE	-1.539167	D	Abengibre
2	CALLE FEDERICO GARCIA LORCA, 1	L-D: 24H	39.000886	ALBACETE	-1.849833	D	Albacete
3	CALLE FEDERICO GARCIA LORCA, 5	L-S: 05:00-23:00	38.99772	ALBACETE	-1.846361	I	Albacete
4	AVENIDA 1º DE MAYO, S/N	L-S: 08:00-22:00 D: 09:00-21:00	38.98567	ALBACETE	-1.868500	N	Albacete
5	CALLE PRINCIPE DE ASTURIAS (POLÍGONO DE ROMICA), 5	L-D: 06:00-21:30	39.05469	ALBACETE	-1.832000	I	Albacete
6	AVENIDA ESCRITOR RODRIGO RUBIO, 3	L-S: 09:00-21:30	39.00689	ALBACETE	-1.885361	D	Albacete
7	AVENIDA MENÉNDEZ PIDAL, 58	L-D: 24H	39.00333	ALBACETE	-1.864917	N	Albacete
8	PASEO CUBA (LA), 36	L-D: 06:00-23:00	39.00508	ALBACETE	-1.859917	N	Albacete
9	CL PASEO DE LA CUBA, 15	L-D: 06:00-22:00	38.99772	ALBACETE	-1.854556	D	Albacete
10	CALLE HERMANOS FALCÓ, 2	L-D: 06:30-22:30	38.98925	ALBACETE	-1.849028	D	Albacete
11	AVDA DE LOS TOREROS, S/N	L-D: 07:00-23:00	38.99822	ALBACETE	-1.869889	N	Albacete
12	CALLE CTRA. DE JAÉN, 79	L-D: 06:00-23:00	38.98811	ALBACETE	-1.883694	I	Albacete
13	PG CAMPOLLANO-P. 2-CT, S.N.	L-V: 06:00-22:00 S: 07:30-15:30; D: 09:00-15:00	39.01258	ALBACETE	-1.874972	D	Albacete

Showing 1 to 13 of 11,273 entries. 32 total columns

Console Terminal Jobs

R 4.1.2 - ~

```

rotulo = col_character(),
tipo_venta = col_character(),
id_provincia = col_character(),
id_ccaa = col_character()
)
1 Use `spec()` for the full column specifications.

```

Imagen 5. Comprobación de cambios de formatos

Al hacer View lo que hago es comprobar que los formatos están correctamente. Ejemplo de latitud. Un problema que me surge es que no em cambia puntos por comas, lo corrojo, pero en el siguiente apartado cuando me toca hacer la media me da error con las comas por lo que decido dejar los puntos.

Todos los archivos
están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

Dirección

lista_eess.precio.Horario

lista_eess.precio.Latitud

lista_eess.precio.Localidad

lista_eess.precio.Longitud (WGS84)

lista_eess.precio.Margen

Showing 1 to 12 of 11,283 entries, 35 total columns

```
R 4.1.2 · ~
<BINARY BODY>
> ds <- jsonlite::fromJSON(url)
> View(ds)
> ds <- ds %>% as_tibble()
> View(ds)
> ds <- ds %>% as_tibble() %>% clean_names()
> View(ds)
> |
```

Imagen 6. Imagen orientativa de como quedaría con comas, aunque decidí dejar puntos.

ii. genere un informe y explique si encuentra alguna anomalía, en el punto ii.

Las anomalías que encuentro son las corregidas en el punto i. La mayoría de las columnas estaban en formato carácter sin serlo, así que he corregido con el clear names y las he puesto en el formato correcto como es latitud. Luego otro de los problemas es el punto por las comas, así que sustituí el punto por las comas y aunque logré corregirlo decidí finalmente dejar el punto porque me daba problemas cuando quería calcular la media.. Este punto se detalla también en el apartado anterior (punto i).

iii. cree una columna nueva que deberá llamarse low-cost, y determine cuál es el precio promedio de todos los combustibles a nivel comunidades autónomas, así como para las provincias, tanto para el territorio peninsular e insular, esta columna deberá clasificar las estaciones por lowcost y no low cost,

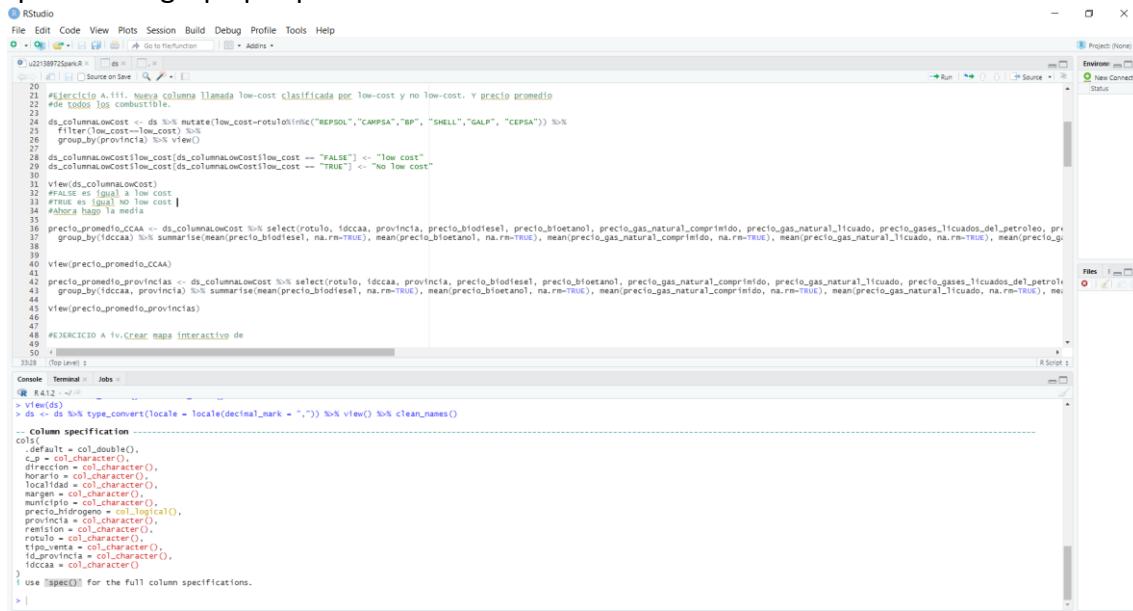
Explicación código: creo un nuevo dataset que es en realidad “copia del anterior” que he llamado ds, este nuevo se llama ds_columnaLowCost porque es donde añado la columna nueva que me pide el enunciado clasificado las gasolineras por low cost y No low cost. Le digo que todo lo que esté en la columna rotulo que sea Repsol, Campsa, BP, Shell, Galp Y Cepsa para mi No Low Cost. Esa sentencia me crea una nueva columna que me devuelve valores TRUE o FALSE. TRUE representa las No low cost y FALSE representa la low cost. Lo que hago es corregir y renombrar esos valores. Y luego calculo precio medio por CCAA y por provincia.

Todos los archivos

están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

Para el precio promedio por CCAA y por provincia lo que hago es en select meter todas las variables a utilizar, le pido que me lo agrupe en el primer caso para CCAA por la variable idccaa y con summarise calculo la media pidiendo que me haga la media de cada variable, añado la función na.rm = TRUE para que no tengo en cuenta los valores na y me realicen correctamente la media.

En el precio promedio por provincias difiere del anterior porque en el group by le digo que me lo agrupe por provincias.

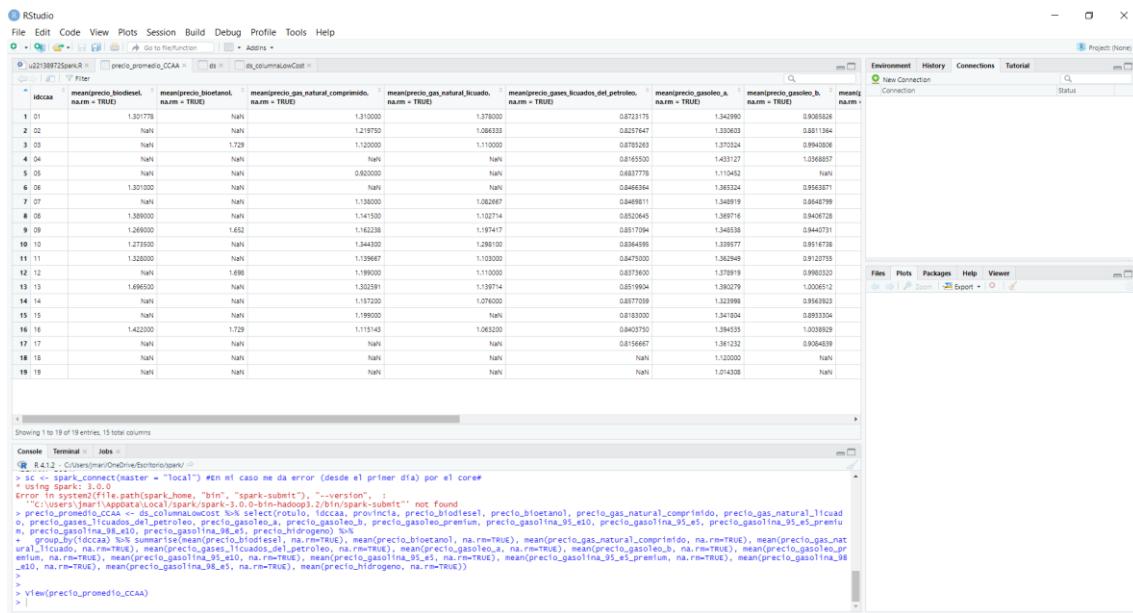


```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
File -> Open... Source on Screen Search <-
21 #Ejercicio A.iii. Nueva columna llamada Tow-cost clasificada por Low-cost y no Tow-cost, y precio promedio
22 #de todos los combustibles
23
24 ds_lowCost <- ds %>% mutate(tow_cost=rotulo[match("REPSOL","CAMPASA","BP","SHELL","GALP","CEPSA")]) %>%
25 filter(tow_cost==low_cost) %>%
26 group_by(provincia) %>% view()
27
28 ds_lowCost$lowCost(ds_lowCost$low_cost == "FALSE") <- "Low cost"
29 ds_lowCost$lowCost(ds_lowCost$low_cost == "TRUE") <- "No low cost"
30
31 V�(ds_lowCost)
32 #V�(ds_lowCost) #que es igual al low cost
33 #true es igual al low cost
34 #ahora hago la media
35
36 precio_promedio_CCAA <- ds_lowCost %>% select(rotulo, idccaa, provincia, precio_biodiesel, precio_bioetanol, precio_gas_natural_comprimido, precio_gas_natural_llicuado, precio_gases_llicuidos_del_petroleo, precio_gases_naturales_llicuados, precio_gases_naturales_llicuado, precio_gases_llicuidos_del_petroleo, precio_gases_naturales_llicuado, precio_gases_naturales_llicuado, precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_biodiesel, na.rm=TRUE), mean(precio_bioetanol, na.rm=TRUE), mean(precio_gas_natural_comprimido, na.rm=TRUE), mean(precio_gas_natural_llicuado, na.rm=TRUE), mean(precio_gases_llicuidos_del_petroleo, na.rm=TRUE), mean(precio_gases_naturales_llicuados, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE))
37 group_by(idccaa) %>% summarise(mean(precio_biodiesel, na.rm=TRUE), mean(precio_bioetanol, na.rm=TRUE), mean(precio_gas_natural_comprimido, na.rm=TRUE), mean(precio_gas_natural_llicuado, na.rm=TRUE), mean(precio_gases_llicuidos_del_petroleo, na.rm=TRUE), mean(precio_gases_naturales_llicuados, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE))
38
39 V�(precio_promedio_CCAA)
40
41
42 precio_promedio_provincias <- ds_lowCost %>% select(rotulo, idccaa, provincia, precio_biodiesel, precio_bioetanol, precio_gas_natural_comprimido, precio_gas_natural_llicuado, precio_gases_llicuidos_del_petroleo, precio_gases_naturales_llicuados, precio_gases_naturales_llicuado, precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_biodiesel, na.rm=TRUE), mean(precio_bioetanol, na.rm=TRUE), mean(precio_gas_natural_comprimido, na.rm=TRUE), mean(precio_gas_natural_llicuado, na.rm=TRUE), mean(precio_gases_llicuidos_del_petroleo, na.rm=TRUE), mean(precio_gases_naturales_llicuados, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE), mean(precio_gases_naturales_llicuado, na.rm=TRUE))
43
44 V�(precio_promedio_provincias)
45
46
47 #EJERCICIO A iv.Crear mapa interactivo de
48
49
50 <
3328 (Top Level)

```

Imagen 7. Código utilizado para el apartado aiii.



idccaa	mean(precio_biodiesel, na.rm = TRUE)	mean(precio_bioetanol, na.rm = TRUE)	mean(precio_gas_natural_comprimido, na.rm = TRUE)	mean(precio_gas_natural_llicuado, na.rm = TRUE)	mean(precio_gases_llicuidos_del_petroleo, na.rm = TRUE)	mean(precio_gases_naturales_llicuados, na.rm = TRUE)	mean(precio_gases_naturales_llicuado, na.rm = TRUE)	mean(precio_gases_llicuidos_del_petroleo, na.rm = TRUE)
1_01	1.301778	NaN	1.310000	1.378000	0.8723175	1.342990	0.908562	0.8723175
2_02	NaN	NaN	1.219750	1.086333	0.8237647	1.330603	0.8811364	0.8237647
3_03	NaN	1.729	1.120000	1.110000	0.8239263	1.370324	0.9404006	0.8239263
4_04	NaN	NaN	NaN	NaN	0.8165500	1.43127	1.0266857	0.8165500
5_05	NaN	0.920000	NaN	NaN	0.8477778	1.110432	NaN	0.8477778
6_06	1.301000	NaN	NaN	NaN	0.8466364	1.363324	0.9563071	0.8466364
7_07	NaN	NaN	1.138000	1.082667	0.8498011	1.348919	0.8648799	0.8498011
8_08	1.389000	NaN	1.145000	1.102714	0.8520645	1.369716	0.9406728	0.8520645
9_09	1.266000	1.652	1.162238	1.197417	0.8517094	1.346538	0.9404071	0.8517094
10_10	1.273000	NaN	1.344300	1.286100	0.8544985	1.339977	0.9116738	0.8544985
11_11	1.328000	NaN	1.199667	1.103000	0.8475000	1.362949	0.9120755	0.8475000
12_12	NaN	1.698	1.199000	1.110000	0.8379600	1.378919	0.9605220	0.8379600
13_13	1.696500	NaN	1.305591	1.190714	0.8519904	1.360279	1.006512	0.8519904
14_14	NaN	NaN	1.197200	1.076000	0.8377059	1.322998	0.9563093	0.8377059
15_15	NaN	NaN	1.199500	NaN	0.8183300	1.347804	0.8933504	0.8183300
16_16	1.422000	1.729	1.115148	1.063200	0.8403750	1.394535	1.0389829	0.8403750
17_17	NaN	NaN	NaN	NaN	0.8156667	1.361232	0.964639	0.8156667
18_18	NaN	NaN	NaN	NaN	NaN	1.120000	NaN	1.120000
19_19	NaN	NaN	NaN	NaN	NaN	1.014308	NaN	1.014308

Imagen 8. Precio medio calculado por CCAA (19 CCAA)

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

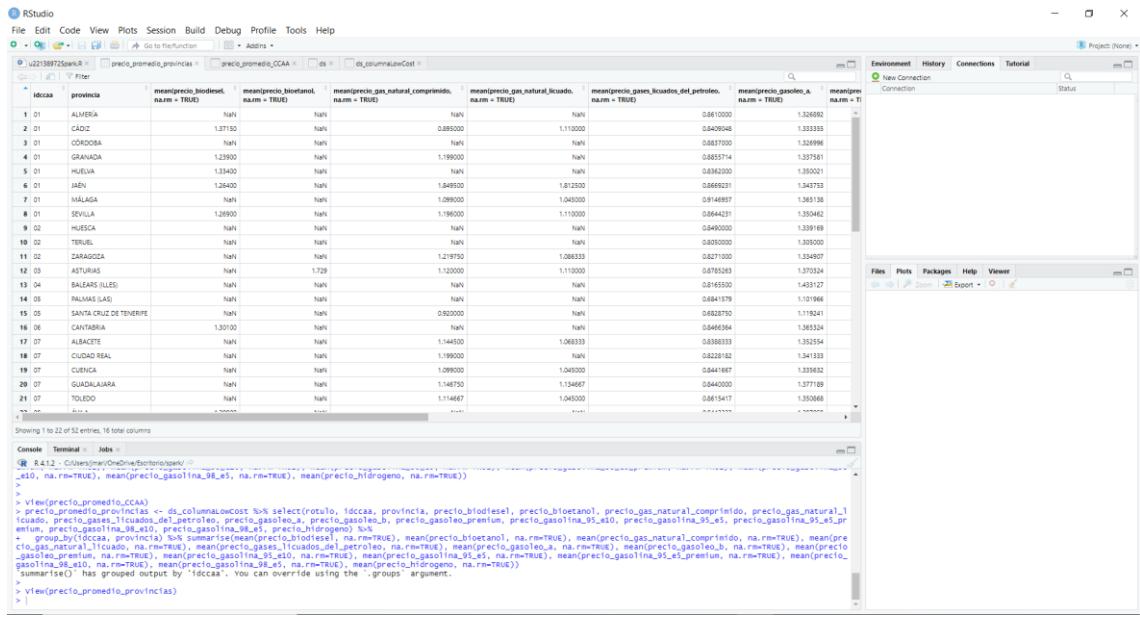


Imagen 9. Precio medio calculado para un total de 52 Provincias.

iv. **Imprima en un mapa interactivo, la localización del top 10 más caras y otro mapa interactivo del top 20 más baratas, estos 2 archivos deben guardarse en formato HTML y pdf para su posterior entrega al inversor., nombre de los archivos: top_10.html, top_10.pdf y top_20.html, top_20.pdf**

Como no se especifica para que tipo de combustible hacerlo. Elijo gasolina 95 e5 Premium ya que es de las más utilizadas en clase a modo explicativo. El código que utilice el aprendido en clase. En select las variables que voy a utilizar, en top_n le digo las diez mas caras que lo indico con numero positivo, en el caso de las 20 más baratas pongo -20, con leaflet es la librería que me ayudar a crear el mapa interactivo, con addTiles me carga la capa base y con addCircleMarkers añado marcas de circulo al mapa.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

Imagen 10. Código utilizado para el apartado aiv.

Lo mas se encuentran disponible en el repositorio según formatos indicado y quedan de la siguiente forma:

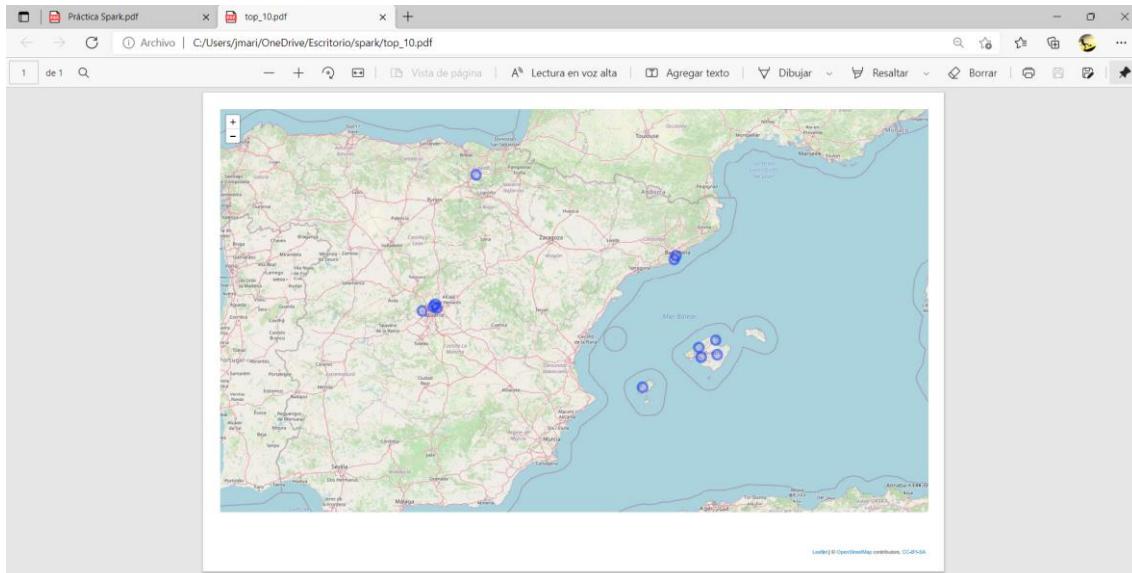


Imagen 11. Mapa interactivo top 10 mas caras para gasolina 95 e5 Premium

Todos los archivos están disponibles:<https://github.com/MarisolGarcialopez?tab=repositories>

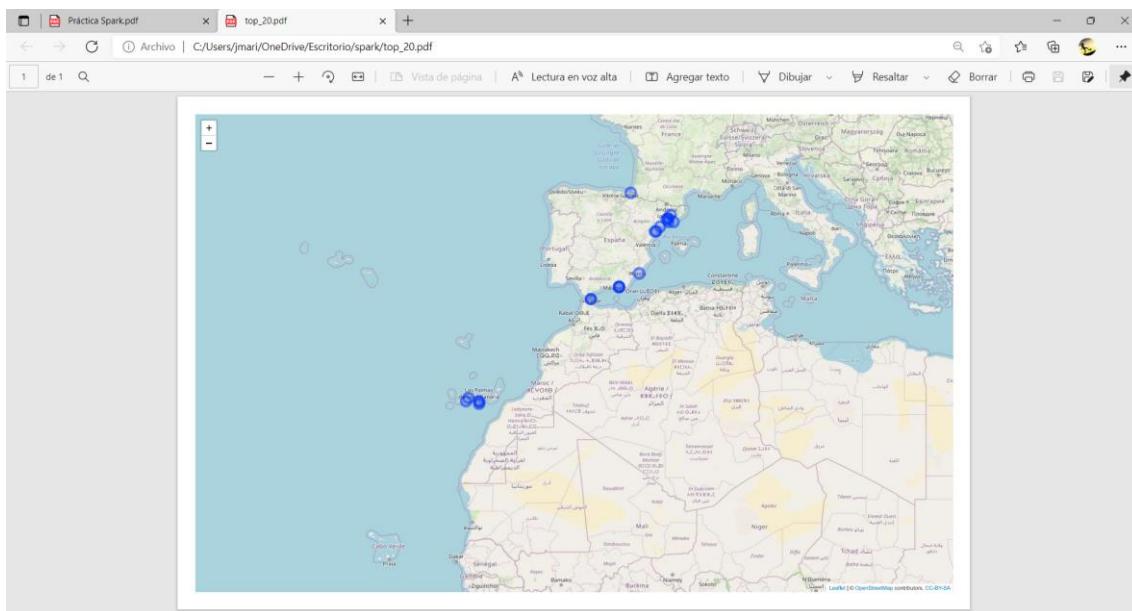


Imagen 12. Top 20 mas baratas para gasolina 95 e5 Premium

v. **conseguidos objetivos anteriores, debe guardar este “archivo” en una nueva tabla llamada low-cost_num_expediente y deberá estar disponible también en su repositorio de Github con el mismo nombre y formato csv.**
Disponible en repositorio según formato indicado.

b. Este empresario tiene sus residencias habituales en Madrid y Barcelona, por lo que, en principio le gustaría implantarse en cualquiera de las dos antes citadas, y para ello quiere saber:

i. **cuántas gasolineras tiene la comunidad de Madrid y en la comunidad de Cataluña, cuántas son low-cost, cuantas no lo son,**

Explicación código: lo hago por separado porque no se como unir ambas comunidades en un mismo código así que opte por hacerlo por separado. Para la comunidad de Madrid sigo trabajando con mi ds_ColumnaLowCost en select meto las variables de idccaa, lowcost y provincia le digo que me lo agrupe con group by por idccaa y provincia y que en el caso de Madrid me lo filtre por idccaa == 13 ya que compruebo que el numero 13 pertenece a la comunidad de Madrid y finalmente que me haga el count de la columna low cost, para clasificarlas. Para Cataluña identifico que es el 09, y el error que me daba es que 09 me lo coge solo con comillas “09” a diferencia del número 13. Siempre finalizo los códigos con un View para comprar el resultado.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

```

# EJERCICIO A V. Guardar archivo en formato CSV
write.csv(ds_columnalowCost, "C:/users/jmari/onedrive/Escritorio/spark/low-cost_u_22138972.csv", row.names = FALSE)

# EJERCICIO B i.
numero_madrid <- ds_columnalowCost %>% select(idccaa, low_cost, provincia) %>% group_by(idccaa, provincia) %>%
filter(idccaa== 13) %>% count(low_cost)
View(numero_madrid)

numero_cataluña <- ds_columnalowCost %>% select(idccaa, low_cost, provincia) %>% group_by(idccaa, provincia) %>%
filter(idccaa== "09") %>% count(low_cost)
View(numero_cataluña)

# Ejericio B ii. Precio promedio, Mas bajo y mas caro de gasoleo a y gasolina 95 e5 Premium
madrid <- ds_columnalowCost %>% select(idccaa, low_cost, provincia, precio_gasoline_95_e5_premium) %>% group_by(idccaa) %>%
filter(idccaa== 13) %>% summarise(mean(precio_gasoline_a, na.rm = TRUE), mean(precio_gasoline_95_e5_premium, na.rm = TRUE), max(precio_gasoline_a, na.rm = TRUE), max(precio_gasoline_95_e5_premium, na.rm = TRUE))
View(madrid)

# Ejericio B iii. Precio promedio, Mas bajo y mas caro de gasoleo a y gasolina 95 e5 Premium
precio_promedio_ccaa <- ds_columnalowCost %>% select(rotulo, idccaa, provincia, precio_biodiesel, precio_biotanol, precio_gas_natural_comprimido, precio_gas_natural_licuado, precio_gases_líquidos_del_petroleo, precio_gasoleo_a, precio_gasoleo_b, precio_gasoline_95_e5_premium, precio_gasoline_95_e10, precio_gasolina_95_e5, precio_gasolina_95_e5_premium, precio_gasolina_98_e10, precio_gasolina_98_e5, precio_hidrogeno) %>% group_by(idccaa, provincia) %>% summarise(mean(precio_biodiesel, na.rm=TRUE), mean(precio_biotanol, na.rm=TRUE), mean(precio_gas_natural_comprimido, na.rm=TRUE), mean(precio_gas_natural_licuado, na.rm=TRUE), mean(precio_gases_líquidos_del_petroleo, na.rm=TRUE), mean(precio_gasoleo_a, na.rm=TRUE), mean(precio_gasoleo_b, na.rm=TRUE), mean(precio_gasoline_95_e10, na.rm=TRUE), mean(precio_gasoline_95_e5_premium, na.rm=TRUE), mean(precio_gasoline_98_e10, na.rm=TRUE), mean(precio_gasoline_98_e5, na.rm=TRUE)), mean(precio_gasoline_a, na.rm=TRUE), mean(precio_gasoline_95_e5_premium, na.rm=TRUE), mean(precio_gasoline_98_e10, na.rm=TRUE), mean(precio_gasoline_98_e5, na.rm=TRUE))
View(precio_promedio_ccaa)

# Ejericio B iv. Precio promedio, Mas bajo y mas caro de gasoleo a y gasolina 95 e5 Premium
ds_columnalowCost %>% select(rotulo, idccaa, provincia, precio_biodiesel, precio_biotanol, precio_gas_natural_comprimido, precio_gas_natural_licuado, precio_gases_líquidos_del_petroleo, precio_gasoleo_a, precio_gasoleo_b, precio_gasoline_95_e5_premium, precio_gasoline_98_e10, precio_gasolina_95_e5, precio_gasolina_95_e5_premium, precio_gasolina_98_e5, precio_hidrogeno) %>% group_by(idccaa, provincia) %>% summarise(mean(precio_biodiesel, na.rm=TRUE), mean(precio_biotanol, na.rm=TRUE), mean(precio_gas_natural_comprimido, na.rm=TRUE), mean(precio_gas_natural_licuado, na.rm=TRUE), mean(precio_gases_líquidos_del_petroleo, na.rm=TRUE), mean(precio_gasoleo_a, na.rm=TRUE), mean(precio_gasoleo_b, na.rm=TRUE), mean(precio_gasoline_95_e10, na.rm=TRUE), mean(precio_gasoline_95_e5_premium, na.rm=TRUE), mean(precio_gasoline_98_e10, na.rm=TRUE), mean(precio_gasoline_98_e5, na.rm=TRUE)), mean(precio_gasoline_a, na.rm=TRUE), mean(precio_gasoline_95_e5_premium, na.rm=TRUE), mean(precio_gasoline_98_e10, na.rm=TRUE), mean(precio_gasoline_98_e5, na.rm=TRUE))
summarise() has grouped output by 'idccaa'. You can override using the '.groups' argument.
View(numero_madrid)

```

Imagen 13. Código utilizado para apartado Bi.

Para la comunidad de Madrid hay un total de 318 gasolineras low cost y 474 que son no low cost.

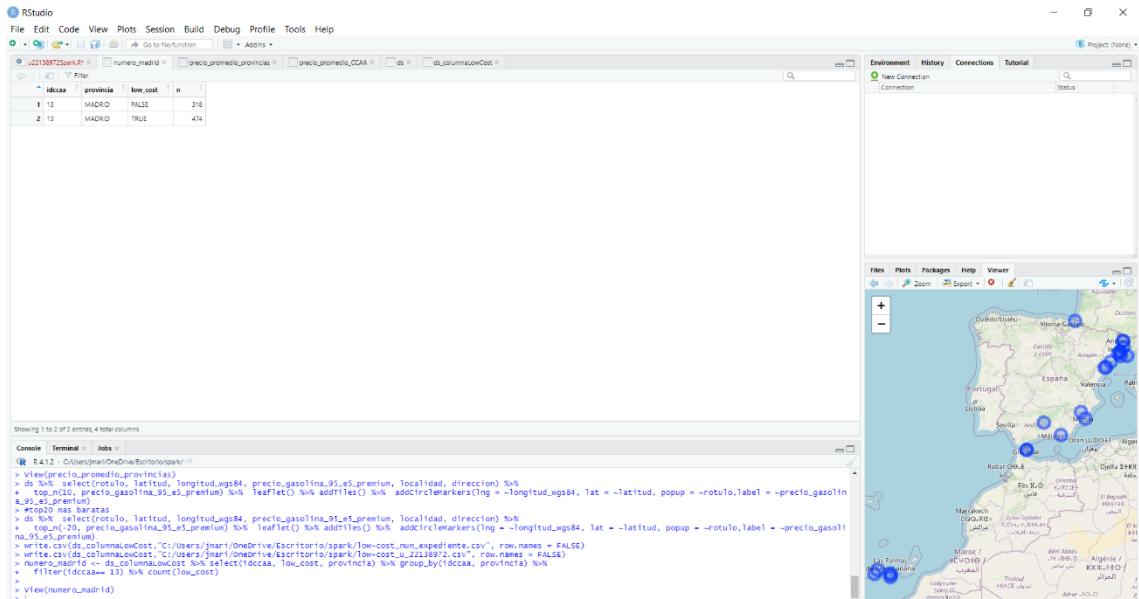


Imagen 14. Total gasolineras Madrid clasificadas como Low-cost y No low-cost

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

En Cataluña hay un total de 820 gasolineras low cost y 654 que son No low cost.

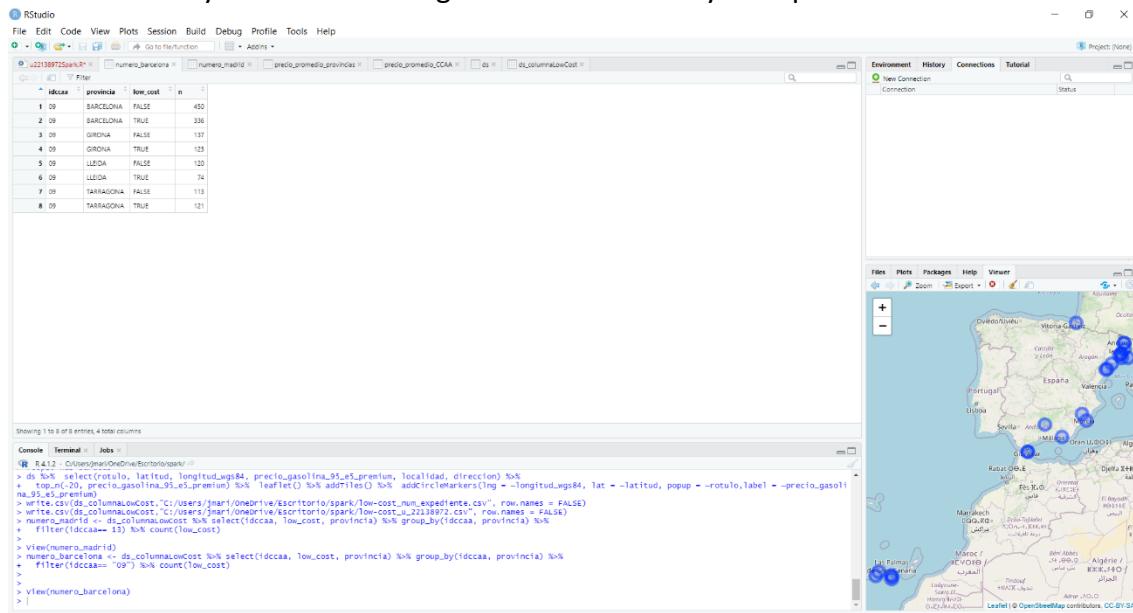


Imagen 15. Total de gasolineras Cataluña clasificadas como Low-cost y No low-cost

ii.además, necesita saber cuál es el precio promedio, el precio más bajo y el más caro de los siguientes carburantes: gasóleo A, y gasolina 95 e Premium.

Explicación código: Aquí lo que hago es añadir a mi sentencia tanto el precio de gasoleo a y de gasolina 95 e5 premium en mi select. Lo agrupo por idccaa ya que solo busco para comunidad de Madrid y Cataluña. Y le agrego un summarise para las operaciones de mean, max, y min. También agrego a todas estas operaciones el na.rm para los valores Na y que no me de fallo al momento de calcularlos.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
i u22138972SpnR [ ] precio_promedio_provincias precio_promedio_CCAA [ ] ds [ ] ds_columnalowCost [ ]
57
58
59 #EJERCICIO A -> Guardar archivo en formato CSV
60
61 write.csv(ds_columnalowCost,"C:/Users/jmarl/OneDrive/Escritorio/spark/low-cost_u_22138972.csv", row.names = FALSE)
62
63
64 #EJERCICIO B -
65 numero_madrid <- ds_columnalowCost %>% select(idccaa, low_cost, provincia) %>% group_by(idccaa, provincia) %>%
66 filter(idccaa == 13) %>% count(low_cost)
67
68 View(numero_madrid)
69
70 numero_cataluha <- ds_columnalowCost %>% select(idccaa, low_cost, provincia) %>% group_by(idccaa, provincia) %>%
71 filter(idccaa == 13) %>% count(low_cost)
72
73
74 View(numero_cataluha)
75
76
77 #Ejercicio B.1. Precio promedio. Mas bajo y mas caro de gasoleo a y gasolina 95 e5 Premium
78
79 madrid <- ds_columnalowCost %>% select(idccaa, low_cost, provincia, precio_gasoleo_a, precio_gasolina_95_e5_premium) %>% group_by(idccaa) %>%
80 filter(idccaa == 13) %>% summarise(mean(precio_gasoleo_a, na.rm = TRUE), mean(precio_gasolina_95_e5_premium, na.rm = TRUE), max(precio_gasoleo_a, na.rm = TRUE), min(precio_gasoline_95_e5_premium, na.rm = TRUE))
81
82
83 View(madrid)
84
85
86 barcelona <- ds_columnalowCost %>% select(idccaa, low_cost, provincia, precio_gasoleo_a, precio_gasolina_95_e5_premium) %>% group_by(idccaa) %>%
87 filter(idccaa == 13) %>% summarise(mean(precio_gasoleo_a, na.rm = TRUE), mean(precio_gasolina_95_e5_premium, na.rm = TRUE), max(precio_gasoleo_a, na.rm = TRUE), min(precio_gasoline_95_e5_premium, na.rm = TRUE))
88
89 View(barcelona)
90
91
92

```

Imagen 16. Código utilizado en el Bii.

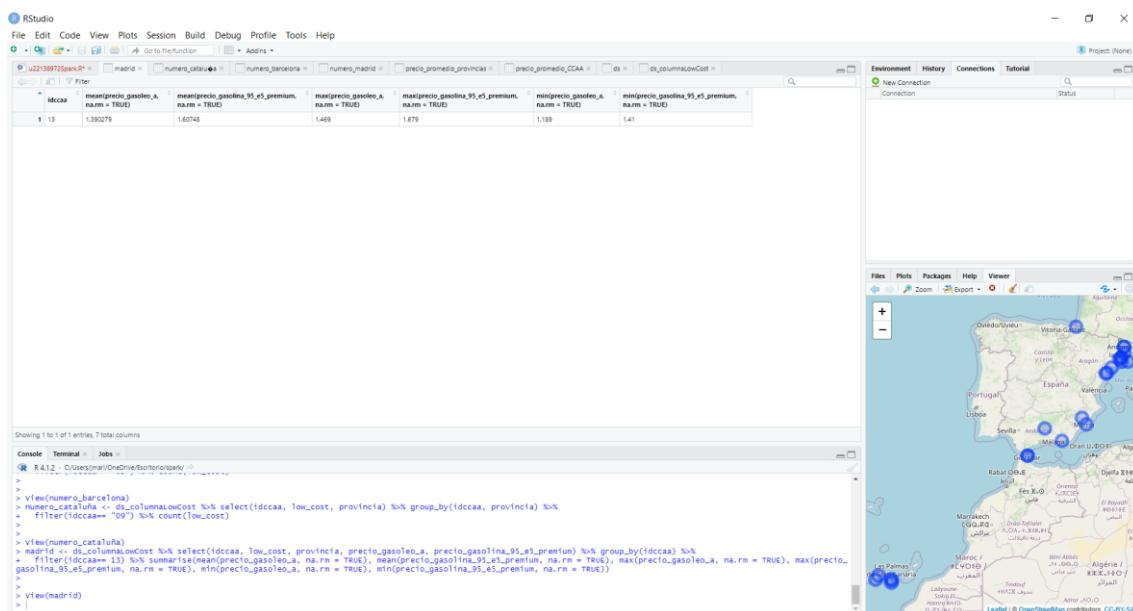


Imagen 17. Precio promedio, Max y min Gasoleo A y Gasolina 95 e5 Premium para Comunidad de MADRID.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

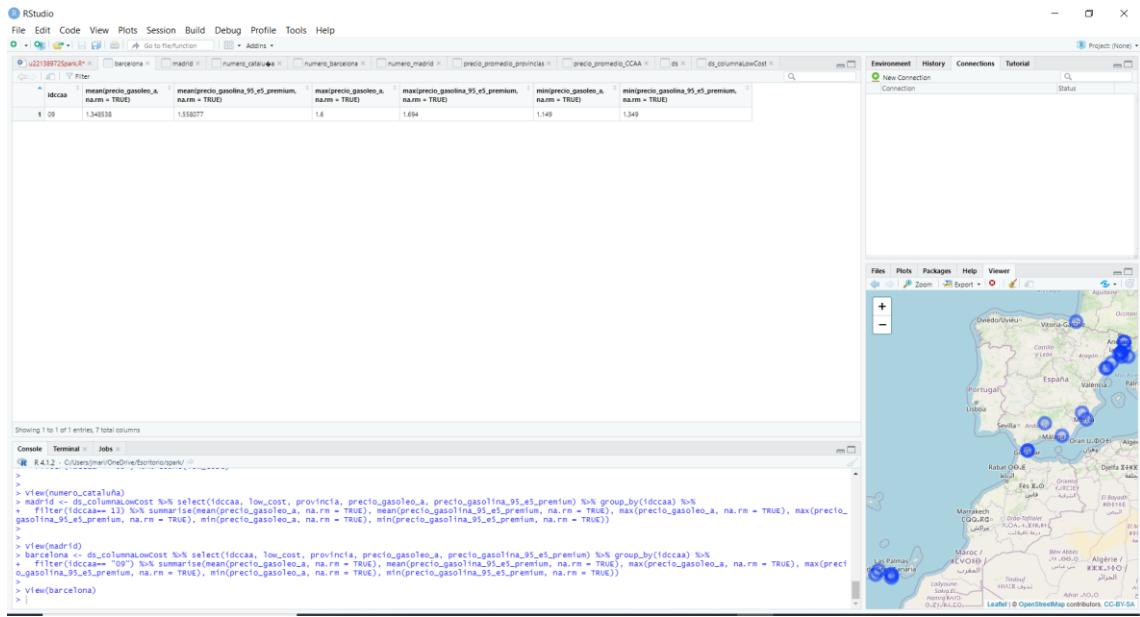


Imagen 18. Precio promedio, Max y min para Gasoleo A y Gasolina 95 e5 premium para Cataluña

iii. Conseguido el objetivo, deberá guardar este “archivo” en una nueva tabla llamada informe_MAD_BCN_expediente y deberá estar disponible también en su repositorio con el mismo nombre en formato csv

Nota: guardo en dos archivos por separado llamadas informe_MAD_miexpediente y informe_BCN_miexpediente, ya que no consigo meterlos en un mismo archivo. Disponibles en el repositorio.

c. Por si las comunidades de Madrid y Cataluña no se adapta a sus requerimientos, el empresario también quiere :

i. **conocer a nivel municipios, cuántas gasolineras son low-cost, cuantas no lo son, cuál es el precio promedio, el precio más bajo y el más caro de los siguientes carburantes: gasóleo A, y gasolina 95 e5 Premium, en todo el TERRITORIO NACIONAL, exceptuando las grandes CIUDADES ESPAÑOLAS ("MADRID", "BARCELONA", "SEVILLA" y "VALENCIA")**

Explicación código: aquí añado municipio y id_municipio tanto en select, para luego en el group by, agruparlo por municipio y en el filtro le pongo el signo de exclamación ! para que no tener en cuenta Madrid, Barcelona, Sevilla y Valencia y en el summarise calculo el precio promedio, precio mínimo y precio máximo.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

```

U22138972sparkR | barcelona | madrid | numero_cataluña | numero_madrid | precio_promedio_provincias | precio_promedio_CCAA | ds | ds_columnalowCost |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| View(barcelona) |
91 #ejercicio 0.11. Guardar archivo en formato csv
92 write.csv(madrid, "C:/users/jmari/onedrive/Escritorio/spark/informe_MAD_u_22138972.csv", row.names = FALSE)
93 write.csv(barcelona, "C:/users/jmari/onedrive/Escritorio/spark/informe_BCN_u_22138972.csv", row.names = FALSE)
94
95
96 #ejercicio 0.12. A nivel municipios, cuantas gasolineras son low cost y no low cost con precio promedio, mas bajo y mas caro, exceptuando grandes ciudades españolas
97 no_grandes_ciudades <- ds_columnalowCost %> select(idccaa, id_municipio, municipio, low_cost, precio_gasoleo_a, precio_gasolina_95_e5_premium) %>% group_by(municipio, low_cost) %>% filter(!municipio %in% c("Madrid", "Barcelona"))
98 View(no_grandes_ciudades)
99 summarise(max(precio_gasoleo_a), min(precio_gasoleo_a), mean(precio_gasoleo_a), min(precio_gasolina_95_e5_premium), mean(precio_gasolina_95_e5_premium))
100
101 cantidad_grandes_ciudades <- no_grandes_ciudades %>% group_by(low_cost) %>% count(low_cost)
102 View(cantidad_grandes_ciudades)
103 write.csv(cantidad_grandes_ciudades, "C:/users/jmari/onedrive/Escritorio/spark/informe_no_grandes_ciudades_cantidad_u22138972.csv", row.names = FALSE)
104
105
106 #ejercicio 0.14. Gasolineras que se encuentran abiertas 24 h exclusivamente
107 no_24_horas <- ds_columnalowCost %>% select(rotulo, provincia, horario, direccion) %>% group_by(provincia) %>% filter(horario == "L-D: 24H") %>% select(rotulo, provincia, horario, direccion)
108 View(no_24_horas)
109
110 #ejercicio 0.14. Guardar archivo en formato excel
111 library(xlrd)
112 library(xlsx)
113
114 write.xlsx(no_24_horas, "C:/users/jmari/onedrive/Escritorio/spark/no_24_horas_u22138972.xlsx")
115
116
117
118
119
120
121
122
123
124

```

Imagen 19. Código utilizado para el apartado ci.

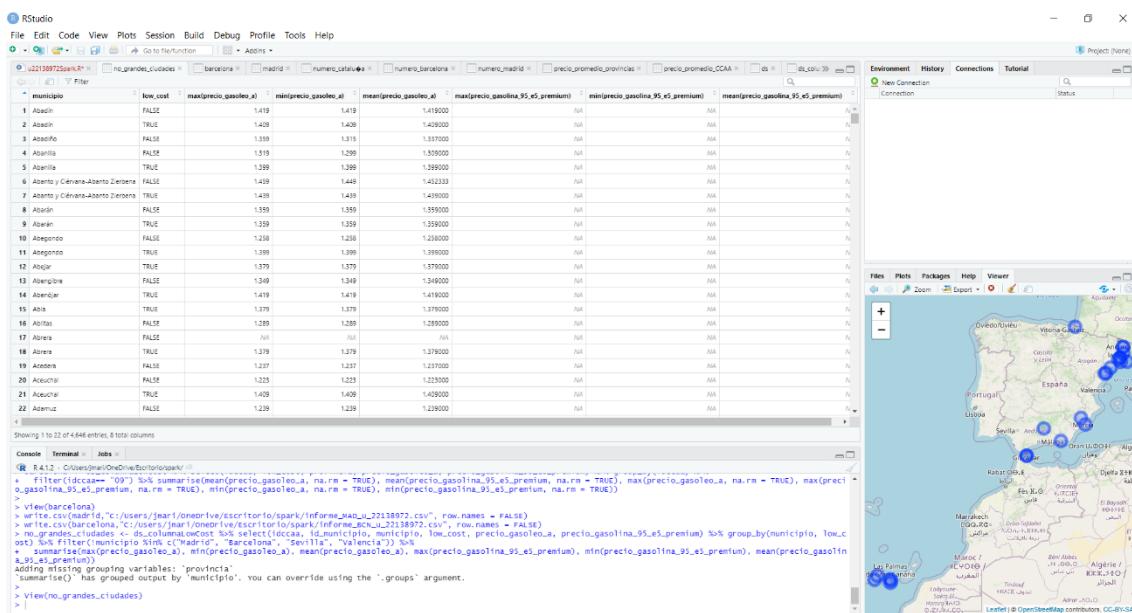


Imagen 20. Precio promedio, mas bajo y caro a nivel municipios exceptuando las grandes ciudades.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

A nivel municipios hay 2480 gasolineras low cost y 2166 no low cost.

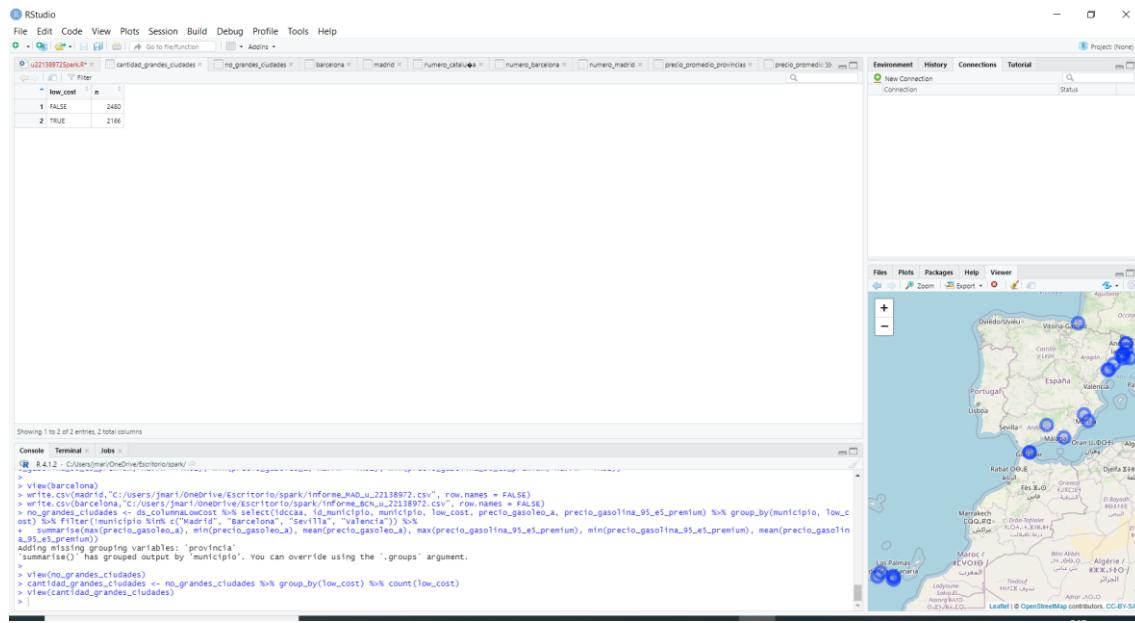


Imagen 21. Cantidad de gasolineras clasificadas como low cost y no low cost a nivel municipio

Para saber cuantas lo que hago es un count por low cost.

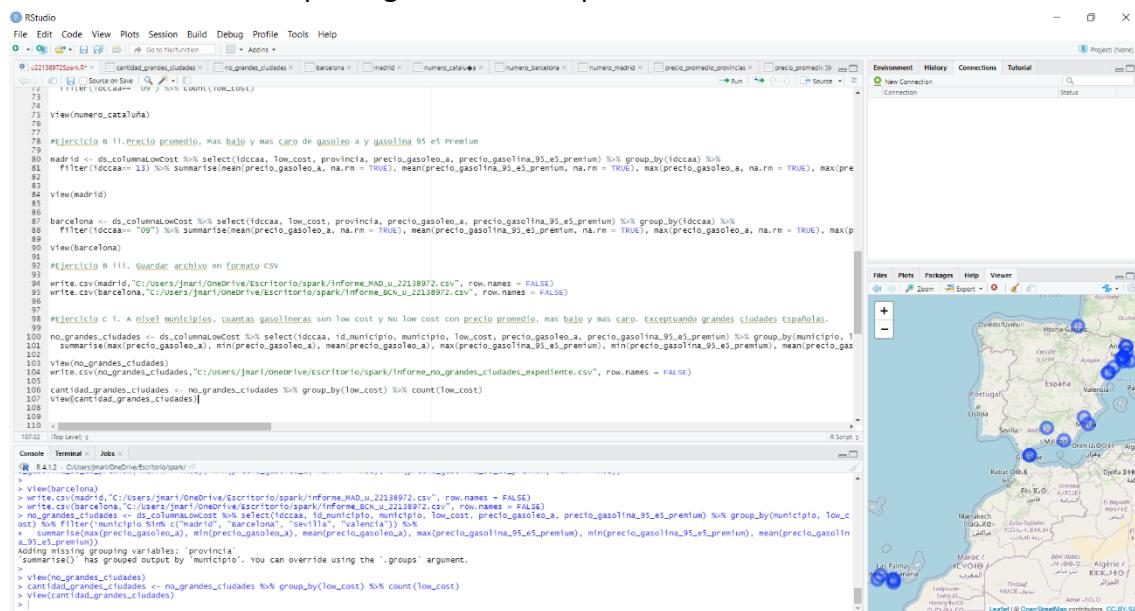


Imagen 22. Código del enunciado ci. Para calcular cuantas gasolineras hay siendo low cost y no low cost.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

ii. *Conseguido el objetivo, deberá guardar este “archivo” en una nueva tabla llamada informe_no_grandes_ciudades_expediente y deberá estar disponible también en su repositorio con el mismo nombre en formato Excel*

En mi caso cuanto intento cargar librería para poder guardar en formato Excel me da error por JAVA. Así que lo guardo en csv y lo convierte a Excel. Y guardo en dos archivos distintos uno para precio promedio, máximo y mínimo a nivel municipios y otro para cantidad clasificadas como low cost y no low cost a nivel municipios.

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
      Go to file/function     Addins

u22138972sun.R cantidad_grandes_ciudades no_grandes_ciudades barcelona madrid numero_cataluna numero_barcelona numero_madrid precio_promedio_provincias precio_promedio_>>
[1] Source on Save [2] Run [3] Source in LFORMAT CSV [4] Run [5] Source [6] Source [7] Source [8] Source [9] Source [10] Source [11] Source [12] Source [13] Source [14] Source [15] Source [16] Source [17] Source [18] Source [19] Source [20] Source [21] Source [22] Source [23] Source [24] Source [25] Source [26] Source [27] Source [28] Source [29] Source [30] Source [31] Source [32] Source [33] Source [34] Source [35] Source [36] Source [37] Source [38] Source [39] Source [40] Source [41] Source [42] Source [43] Source [44] Source [45] Source [46] Source [47] Source [48] Source [49] Source [50] Source [51] Source [52] Source [53] Source [54] Source [55] Source [56] Source [57] Source [58] Source [59] Source [60] Source [61] Source [62] Source [63] Source [64] Source [65] Source [66] Source [67] Source [68] Source [69] Source [70] Source [71] Source [72] Source [73] Source [74] Source [75] Source [76] Source [77] Source [78] Source [79] Source [80] Source [81] Source [82] Source [83] Source [84] Source [85] Source [86] Source [87] Source [88] Source [89] Source [90] Source [91] Source [92] Source [93] Source [94] write.csv(madrid, "C:/users/jmari/onedrive/escritorio/spark/informe_MAD_U_22138972.csv", row.names = FALSE)
[95] write.csv(barcelona, "C:/users/jmari/onedrive/escritorio/spark/informe_BCN_U_22138972.csv", row.names = FALSE)
[96]
[97] #Ejercicio C 1. A nivel municipios, cuantas gasolineras son low cost y no low cost con precio promedio, mas bajo y mas caro. Exceptuando grandes ciudades Españolas.
[98]
[99] no_grandes_ciudades <- ds_columnalowcost %>% select(idccaa, id_municipio, municipio, low_cost, precio_gasoleo_a, precio_gasolina_95_e5_premium) %>% group_by(municipio, 1
[100] 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_gas
[101]
[102] VView(no_grandes_ciudades)
[103] write.csv(no_grandes_ciudades, "C:/users/jmari/onedrive/escritorio/spark/informe_no_grandes_ciudades_u22138972.csv", row.names = FALSE)
[104]
[105] cantidad_grandes_ciudades <- no_grandes_ciudades %>% group_by(low_cost) %>% count(low_cost)
[106] VView(cantidad_grandes_ciudades)
[107] write.csv(no_grandes_ciudades, "C:/users/jmari/onedrive/escritorio/spark/informe_no_grandes_ciudades_cantidad_u22138972.csv", row.names = FALSE)
[108]
[109]
[110] #Ejercicio D 1.
[111]
[112] no_24_horas <- ds_columnalowcost %>% select(rutodo, provincia, horario, direccion) %>% group_by(provincia) %>% filter(horario == "L-D: 24H") %>% select(rutodo, provinci
[113] VView(no_24_horas)
[114]
[115]
[116] #Ejercicio D 11.
[117] library(XML)
[118]
[119] library(xlsx)
[120]
[121] write.csv(no_24_horas, "C:/users/jmari/onedrive/escritorio/spark/no_24_horas.csv", row.names = FALSE)
[122]
[123] #Ejercicio e 1.
[124]
[125] library(spatial)
[126] library(raster)
[127] pobmun21 <- read_csv("pobmun21.csv")
[128] VView(pobmun21)
[129]
[130]
[1211] (Top Level) z
```

Imagen 23. Error que me da al cargar librería xlsx para guarda en formato Excel.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

d. Determine :

i. que gasolineras se encuentran abiertas las 24 horas exclusivamente, genere una nueva tabla llamada no_24_horas sin la variable horario (es decir no debe aparecer esta columna).

Explicación código: aquí hago un doble select, en el primer select incluyo las variables a utilizar y dentro de ellas horario. Le pido que me lo filtre por el horario de las que están abiertas 24 de lunes a domingo y luego vuelvo aplicar otro select con el signo de exclamación para que no me incluya horario.

```

# Ejercicio B 111. Guardar archivo en formato csv
94 write.csv(madrid, "C:/users/jmarl/onedrive/escritorio/spark/informe_MAD_U_22138972.csv", row.names = FALSE)
95 write.csv(barcelona, "C:/users/jmarl/onedrive/escritorio/spark/informe_BCN_U_22138972.csv", row.names = FALSE)
96
97 #Ejercicio C 1. A nivel municipios, cuantas gasolineras son Low cost y No Low cost con precio promedio, mas bajo y mas caro. Exceptuando grandes ciudades Españolas.
98 no_grandes_ciudades <- ds_columnalowCost %>% select(idccaa, low_cost, provincia, precio_gasoleo_a, precio_gasolina_95_e5_premium) %>% group_by(idccaa) %>%
99   filter(idccaa == "09") %>% 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))
100 View(no_grandes_ciudades)
101
102 #Ejercicio C 2. Cantidad de gasolineras abiertas 24H en cada provincia
103 View(no_grandes_ciudades)
104 write.csv(no_grandes_ciudades, "C:/users/jmarl/onedrive/escritorio/spark/informe_no_grandes_ciudades_u22138972.csv", row.names = FALSE)
105
106 cantidad_grandes_ciudades <- no_grandes_ciudades %>% group_by(low_cost) %>% count(low_cost)
107 View(cantidad_grandes_ciudades)
108 write.csv(cantidad_grandes_ciudades, "C:/users/jmarl/onedrive/escritorio/spark/informe_no_grandes_ciudades_cantidad_u22138972.csv", row.names = FALSE)
109
110 #Ejercicio D 1. Gasolineras que se encuentran abiertas 24 H EXCLUSIVAMENTE
111 no_24_horas <- ds_columnalowCost %>% select(rotulo, provincia, horario, direccion) %>% filter(horario == "L-D: 24H") %>% select(rotulo, provincia, horario, direccion)
112 View(no_24_horas)
113
114 #Ejercicio D 2. Guardar archivo en formato excel
115 library(xlsx)
116
117 write.xlsx(no_24_horas, "C:/users/jmarl/onedrive/escritorio/spark/no_24_horas.csv", row.names = FALSE)
118
119 #Ejercicio E 1.
120
121 #Ejercicio E 2.
122
123 #Ejercicio E 3.
124
125 #Ejercicio E 4.
126
127 # (Top Level)

```

Imagen 24. Código utilizado para el enunciado di.

rotulo	provincia	direccion
1 PLENOL	ALBACETE	CALLE FEDERICO GARCIA LORCA, 1
2 TANOS	ALBACETE	AVENIDA MENCHON PIDAL 58
3 REPSOL	ALBACETE	CR N-301, 244
4 REPSOL	ALBACETE	CARRETERA JAVI, KM.2
5 CEPSA	ALBACETE	CARRETERA N 322 KM. 349
6 CORMAR	ALBACETE	CIRAGUASINIEVES KM 022
7 ENERCOM CARBURANTES	ALBACETE	POLIGONO AVILA, 37, ISSUNDA/CIAUTOVA, 69
8 QMOL	ALBACETE	AVENIDA PRIMERA, S/N
9 CEPSA	ALBACETE	POLIGONO CAMPOALANZO, 55
10 AVANZA OIL	ALBACETE	POLIGONO CAMPOALANZO C/F, 13
11 NATURGY ES. GNV ALBACETE	ALBACETE	CALLE C/AUTOMA (ZV. ZETA, SXTA, S/N)
12 VALERO ZAH	ALBACETE	AVENIDA POLIGONO CAMPOALANZO AVENIDA 0, 87
13 PAST & GAS	ALBACETE	AVENIDA DE PESQUINA C/FMS, 3
14 CEPSA	ALBACETE	AUTOMA A-20 KM.239,4
15 GASOLINERA LOW COST PEÑAIDES	ALBACETE	CARRERILLA N-430 RADIANZOS A VALENCIA KM. 594
16 QUALITY ALMANSA	ALBACETE	CALLE INFANTE DON JUAN MANUEL, 4
17 GASOLEOS S ALMANSA	ALBACETE	AVENIDA CRONICA/LA CON, 17
18 COOPERATIVA SANTA CRUZ DE ALPERA	ALBACETE	BONITE, S/N
19 COABA	ALBACETE	CAMINO DE LOS GONDALOS SN
20 CEPSA	ALBACETE	AVDA. DEL VINDO, 13
21 HAM	ALBACETE	CARRERILLA D-12 ESTACION KM,1
22 TRANSPORTES CADETE	ALBACETE	CARRERILLA N-344 KM. 114,500
23 VILA Y FAJOS	ALBACETE	CARRERILLA D-12 ESTACION KM, 32

Imagen 25. Gasolineras abiertas con su dirección abiertas 24H exclusivamente.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

ii. *Conseguido el objetivo, deberá guardar este “archivo” en una nueva tabla llamada no_24_horas y deberá estar disponible también en su repositorio con el mismo nombre en formato Excel*

Al ser en formato Excel me vuelve a pasar lo mismo. Así que guardo el archivo en csv y lo convierto. Disponible en repositorio.

e. Uno de los factores más importantes para que el empresario se decante a instalar nuevas gasolineras es la demanda que viene dada por la población y la competencia existente en un municipio donde se pretenda implantar las gasolineras, para responder a esta pregunta de negocio,

i. *deberá añadir la población al dataset original creando una nueva columna denominada población, esta información debe ser veraz y la más actualizada, la población debe estar a nivel municipal (todo el territorio nacional)*

Aquí utilizo los mismo datos de población que vimos en Hadoop solo que actualizados para 2021. Para eso me voy a la página del INE y en el apartado de demografía y población y padrón encuentro el csv a utilizar. Lo guardo en mi directorio de trabajo y lo cargo con read_csv. Observo los datos y lo primero que quiero lograr es que están igual a mis otros datos, por eso vuelvo a renombrar el titulo de las columnas, corrojo CPRO por id_provincia, cambio las mayúsculas de provincia al igual que las de cmun y nombre por municipio.

A la unión de ambos lo voy a llamar “unión” y utilizo left join a lo que le estoy diciendo que me una mi dataset original que es ds y el que quiero añadir que es población (el que contiene las columnas corregidas) y luego “municipio” que sería mi columna en común.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

```

#> 12138972SparkR <- sparkRSession("pobmun21", "no_24_horas", "cantidad_grandes_ciudades", "no_grandes_ciudades", "barcelona", "madrid", "numero_catujo", "numero_madrid", "precio_promedio_provincias", "precio_promedio_CCAA", "ds", "ds_columnacionCost")
111 #Ejercicio 01. Gasolineras que se encuentran abiertas 24 H exclusivamente
112 no_24_horas <- ds_columnacionCost %>% select(rotulo, provincia, horario, direccion) %>% group_by(provincia) %>% filter(horario == "L=O: 24H") %>% select(rotulo, provincia, horario, direccion)
113 
114 View(no_24_horas)
115 
116 
117 #Ejercicio 01. guardar archivo en formato excel
118 library(XML)
119 library(xlsx)
120 
121 write.csv(no_24_horas, "C:/users/jmari/onedrive/escritorio/spark/no_24_horas_u22138972.csv", row.names = FALSE)
122 
123 #Ejercicio 01. Añadir población al dataset original cuando existentes desvinculados pobmun21
124 library(spatial)
125 
126 View(poblacion)
127 
128 pobmun21 <- read_csv("pobmun21.csv")
129 View(pobmun21)
130 
131 names(pobmun21)
132 
133 poblacion <- rename(pobmun21, id_provincia = "CPRO", provincia = "PROVINCIA", cmun = "CMUN", municipio = "NOMBRE")
134 View(poblacion)
135 
136 union <- left_join(ds, poblacion, "municipio")
137 View(union)
138 
139 #Ejercicio 01. Calcule competencia con gasolinera y dirección
140 union <- left_join(union, rotulo, "rotulo", "localidad, dirección") %>%
141   addin_markers(lng = -longitud_wgs84, lat = -latitud, popup = ~rotulo,label = ~municipio) %>%
142   add_markers(lng = -longitud_wgs84, lat = -latitud, radius = 1000)
143 
144 union <- select(union, rotulo, latitud, longitud_wgs84, municipio, localidad, dirección) %>%
145   add_markers(lng = -longitud_wgs84, lat = -latitud, popup = ~rotulo,label = ~municipio) %>%
146   add_markers(lng = -longitud_wgs84, lat = -latitud, radius = 1000)
147 
148 #> 12138972SparkR <- stop()
149 
150 #> 12138972SparkR <- null()
151 
152 #> 12138972SparkR <- null()

```

Imagen 26. Código utilizado en el apartado ei.

id_gasolina_98_e5	precio_gasolina_98_e5	precio_hidrogeno	provincia.x	remision	rotulo	tipo_venta	percent_bio_etanol	percent_este_metilico	ideess	id_municipio	id_provincia.x	idcaa	id_provincia.y	provincia.y	cmun	POR21
NA	NA	NA	ALBACETE	OM	M 10393	P	0	0	4375	52	02	07	2	Alicante	1	748
NA	NA	NA	ALBACETE	OM	REPSOL	P	0	0	5122	53	02	07	2	Alicante	2	496
NA	1.803	NA	ALBACETE	OM	CARREFOUR	P	0	0	10785	54	02	07	2	Alicante	3	172722
NA	1.639	NA	ALBACETE	OM	CEPSA	R	0	0	4438	54	02	07	2	Alicante	3	172722
NA	NA	NA	ALBACETE	OM	PEÑOL	P	0	0	13833	54	02	07	2	Alicante	3	172722
NA	1.641	NA	ALBACETE	OM	BP ROMICA	P	0	0	12054	54	02	07	2	Alicante	3	172722
NA	1.629	NA	ALBACETE	OM	REPSOL	P	0	0	5195	54	02	07	2	Alicante	3	172722
NA	1.599	NA	ALBACETE	OM	TAMOS	P	0	0	4369	54	02	07	2	Alicante	3	172722
NA	1.599	NA	ALBACETE	OM	CEPSA	P	0	0	14123	54	02	07	2	Alicante	3	172722
NA	NA	NA	ALBACETE	OM	A&A	P	0	0	15000	54	02	07	2	Alicante	3	172722
NA	1.629	NA	ALBACETE	OM	CEPSA	P	0	0	5313	54	02	07	2	Alicante	3	172722
NA	NA	NA	ALBACETE	OM	INPEASLA	P	0	0	4415	54	02	07	2	Alicante	3	172722
NA	1.639	NA	ALBACETE	OM	REPSOL	P	0	0	5116	54	02	07	2	Alicante	3	172722
NA	1.631	NA	ALBACETE	OM	CEPSA	P	0	0	4795	54	02	07	2	Alicante	3	172722
NA	1.619	NA	ALBACETE	OM	REPSOL	P	0	0	5222	54	02	07	2	Alicante	3	172722
NA	1.629	NA	ALBACETE	OM	REPSOL	P	0	0	5245	54	02	07	2	Alicante	3	172722
NA	1.519	NA	ALBACETE	OM	FAMILY ENERGY	P	0	0	4428	54	02	07	2	Alicante	3	172722
NA	1.619	NA	ALBACETE	OM	REPSOL	P	0	0	5208	54	02	07	2	Alicante	3	172722
NA	1.629	NA	ALBACETE	OM	REPSOL	P	0	0	5191	54	02	07	2	Alicante	3	172722
NA	1.619	NA	ALBACETE	OM	REPSOL	P	0	0	5289	54	02	07	2	Alicante	3	172722

Imagen 27. Resultado del left join, View de mi tabla Union.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

ii. este empresario ha visto varios sitios donde potencialmente le gustaría instalar su gasolinera, esos sitios están representados por la dirección, desde esta última calcule cuanta competencia (nombre de la gasolinera y dirección) tiene en:
1. En un radio de 1 km (genera mapa_competencia1.html)
2. En un radio de 2 km (genera mapa_competencia2.html)
3. En un radio de 4 km (genera mapa_competencia3.html)

Al igual que cuando cree el mapa interactivo en el apartado a.iv utilice una sentencia parecida, la diferencia son las variables para utilizar en este caso utilice municipio y no ningún precio. Lo que entiendo es que este apartado no me pide para un sitio (no se me indica una dirección) en concreto ni un precio de gasolina en concreto. Así que, lo hago a nivel nacional, esto se podría filtrar según municipio o lo que se busque.

La diferencia del código es que aquí añado addCircles que me ayuda a añadir círculos a mi mapa. En este caso con las variables longitud y latitud y “radius” para los kilómetros que se pide, 1km, 2km y 4km. Se aprecian mejor con el zoom donde se ven que a mas radio mas grande los círculos.

The screenshot shows the RStudio interface with the following details:

- Code Editor:** The main area displays R code for a spatial analysis. It includes:
 - Reading a CSV file into a dataset named `pobmun21`.
 - Adding a column `poblacion` to the dataset.
 - Creating a spatial object `polymun21` from the dataset.
 - Performing a left join between `polymun21` and `pobmun21` datasets.
 - Calculating competence with `calcule_competencia` and creating markers with `addCircleMarkers`.
 - Setting up a leaflet map with `leaflet` and adding markers with `addTiles`.
 - Adding circles with `addCircles` and setting a radius of 1000 meters.
 - Adding circles with `addCircles` and setting a radius of 2000 meters.
 - Adding circles with `addCircles` and setting a radius of 4000 meters.
- Terminal:** Below the code editor, the terminal window shows the command used to run the script and the output message indicating the column specification was retrieved.

Imagen 28. Código utilizado en el apartado eii. Mapas de competencia utilizando radios de 1km, 2km, 4km.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

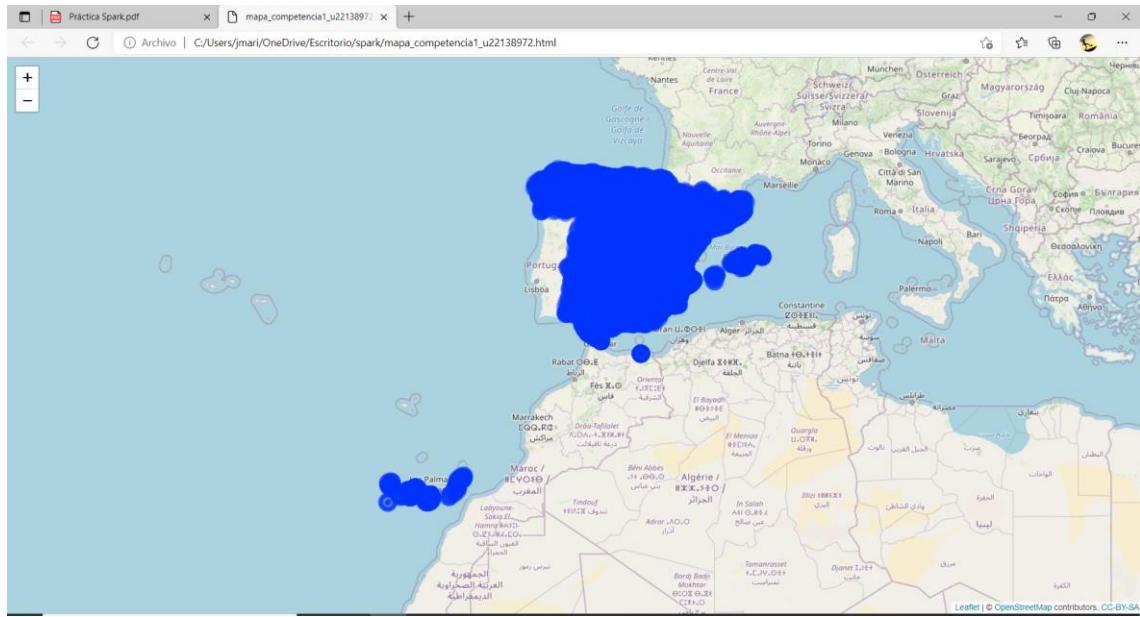


Imagen 29. Mapa de competencia radio de 1km, a nivel nacional.

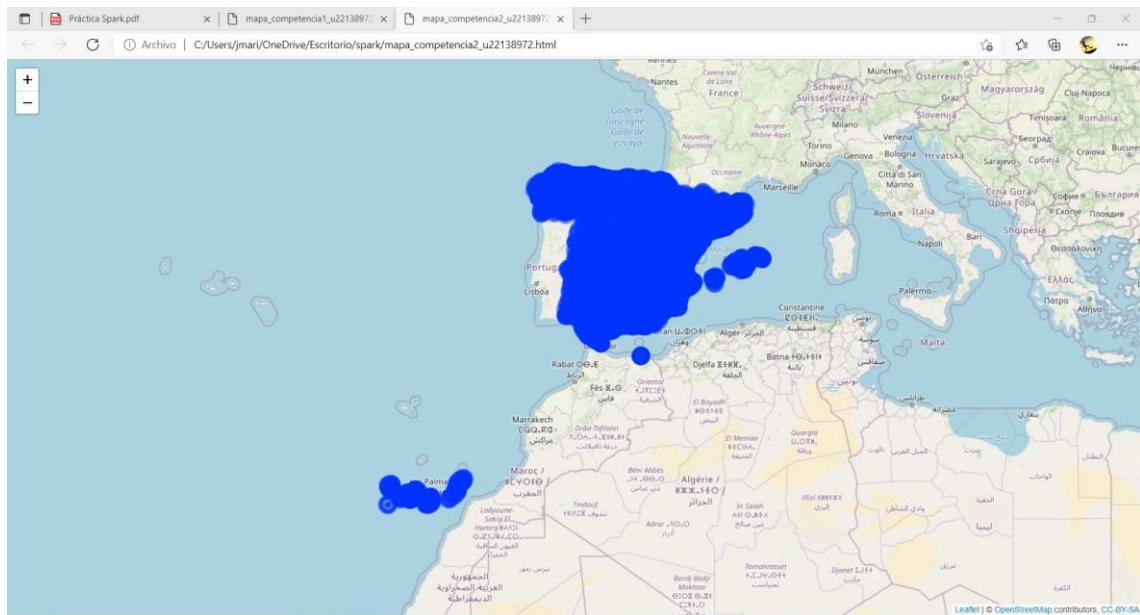


Imagen 30. Mapa de competencia radio 2km, a nivel nacional.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

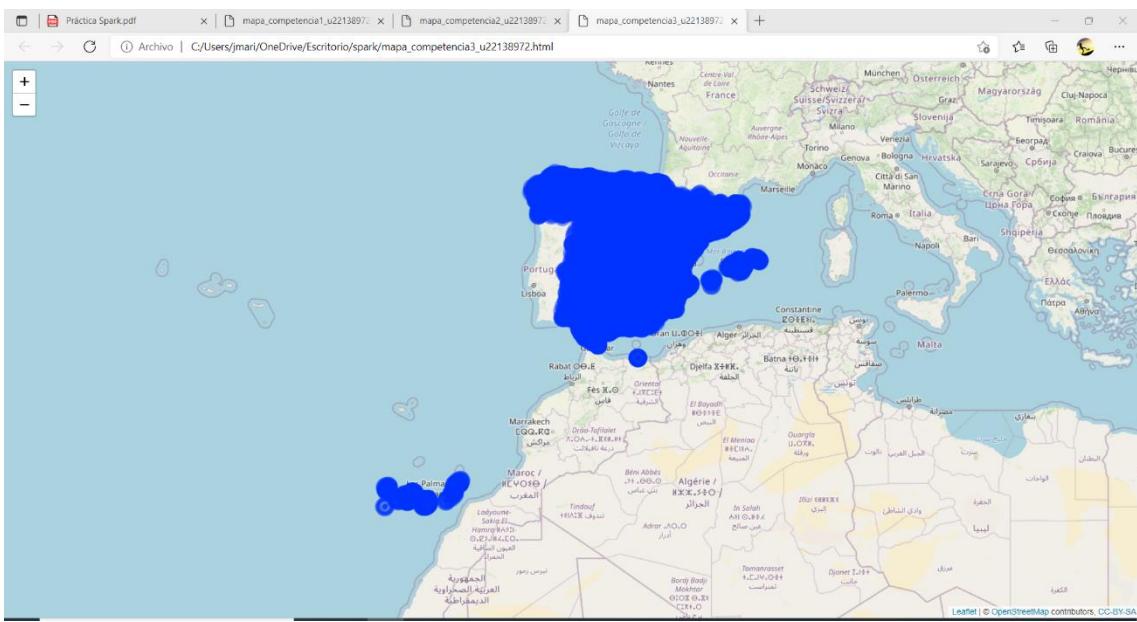


Imagen 31. Mapa de competencia radio 4km, a nivel nacional.

iii. genere el TopTen de municipios entre todo el territorio nacional excepto el territorio insular, donde no existan gasolineras 24 horas, agrupadas entre low-cost y no low-cost, deberá guardar este “archivo” en una nueva tabla llamada informe_top_ten_expediente y deberá estar disponible también en su repositorio con el mismo nombre en formato csv.

Explicación código: en este código vuelvo a utilizar los datos de ds_ColumnaLowCost ya que me pide agruparlas entre low cost y no low cost. Aquí aplico nos filtros el primero porque me dice que no incluya el territorio insular así que le digo filter con signo de exclamación y que en provincia no me incluya el terriotorio insular que es baleares y las palmas. El segundo filter que aplico son las exclusivamente 24 horas, entonces le digo horario == de lunes a domingo que sean 24 horas. Como me pide el top ten de municipios que me la agrupe por municipios y por clasificación de low cost y no low cost. La forma mas rápida e intuitiva de sacar el top ten es que en la columna nueva que me genera con “n” lo orden de mayor a menor, adjunto como quedaría.

Todos los archivos están disponibles:<https://github.com/MarisolGarciaLopez?tab=repositories>

Imagen 32. Código utilizado apartado eiii.

The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help, and Addins. The left sidebar shows a project tree with files like 'u21238972sparkR.R' and 'informe_top_ten.R'. The main workspace displays a data frame 'informe_top_ten' with columns 'municipio', 'low_cost', and 'n'. The data shows 17 rows of Spanish cities and their counts. The bottom-left pane is the 'Console' showing R code execution. The bottom-right pane is a 'Viewer' showing a map of Spain with a blue highlighted area representing the data.

municipio	low_cost	n
1 Madrid	No low cost	75
2 Murcia	No low cost	39
3 Zaragoza	No low cost	39
4 Valencia	No low cost	32
5 Murcia	low cost	31
6 Barcelona	No low cost	30
7 Valladolid	No low cost	30
8 Córdoba	No low cost	25
9 Cartagena	No low cost	24
10 Jerez de la Frontera	No low cost	23
11 Ejido (E)	low cost	20
12 Málaga	No low cost	20
13 Alicante/Alicant	No low cost	18
14 Cartagena	low cost	18
15 Gijón	No low cost	18
16 Sevilla	No low cost	18

Showing 1 to 17 of 3,413 entries. 3 total columns

Console Terminal Jobs

```
R version 4.1.2 -- "Caveat Emptor"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
R is copyrighted free software written by R Core Team.
The license for R is the same as for RStudio: the GNU General Public License version 2 (GPL-2.0+).
R is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
See the GNU General Public License for more details.
You should have received a copy of the GNU General Public License along with R. It should be in /usr/share/licenses/R/GPL-2.0+. If not, see https://www.gnu.org/licenses/.
```

File Environment History Connections Tutorial Project: (None)

Connections

Status

Files Plots Packages Help Viewer

A map of Spain and surrounding regions (Portugal, Andorra, France) is displayed in the 'Viewer' pane. A large blue shaded area covers the eastern part of Spain, including the regions of Aragón, Valencia, and Murcia, as well as parts of the Mediterranean coast. The map also shows the location of the Balearic Islands and the Canary Islands.

Imagen 33. Top ten municipios a nivel nacional exceptuando territorio insular y clasificadas por low cost y no low cost con horario 24 horas.

Todos los archivos
están disponibles: <https://github.com/MarisolGarciaLopez?tab=repositories>