Estudio de la competitividad regional en la Unión Europea utilizando Python

1.

La base de datos proporcionada de la Comisión Europea tiene las siguientes características:

- **Dimensión**: 234 filas y 21 columnas.
- **Columnas**:
    - **REGION**: Nombre de la región (tipo objeto, texto).
    - **NUTS2ID**: Código NUTS2 de la región (objeto, texto).
    - **CNTR**: Código del país (objeto, texto).
    - **DEVLEV**: Nivel de desarrollo de la región (objeto, texto).
    - **RANK**: Rango de la región (entero).
    - **RCI, BSI, INSTIT, MACRO, INFRASTR, HEALTH, BASICEDUC, ESI, HIGHEREDUC, LABMARKET, MARKETSIZE, ISI, TECHREAD, BUSINESSSOPH, INNOV, GDPPC**: Estas variables son valores numéricos que miden distintos indicadores como Competitividad Regional (RCI), Infraestructura (INFRASTR), Mercado Laboral (LABMARKET), PIB per cápita (GDPPC), entre otros.

El código empleado ha sido el siguiente:

```
# Cargar el archivo Excel y revisar las primeras filas
file_path = "C:/Users/November/Documents/RCI_2_0_scores.xlsx"
data = pd.read_excel(file_path, sheet_name=0)

print(data.head())
print(data.info())
```

2.

El análisis descriptivo de las variables cuantitativas del conjunto de datos muestra lo siguiente:

- **RANK**: Rango de la región.
    - Media: 117.35
    - Mínimo: 1, Máximo: 234
    - Percentil 25: 59.25, Percentil 50: 117.5, Percentil 75: 175.75
- **RCI** (Regional Competitiveness Index):
    - Media: 95.9
    - Mínimo: 46.1, Máximo: 151.1
- **BSI** (Basic Subindex):
    - Media: 97.37
    - Mínimo: 37.3, Máximo: 138.1
- **INSTIT** (Institutions):
    - Media: 102.53
    - Mínimo: 34.1, Máximo: 197.9
- **MACRO** (Macroeconomic stability):
    - Media: 101.45
    - Mínimo: 33.6, Máximo: 159.7
- **INFRASTR** (Infrastructure):
    - Media: 85.82
    - Mínimo: 19.5, Máximo: 185.8
- **HEALTH** (Health):
    - Media: 97.05
    - Mínimo: 41.0, Máximo: 127.2
- **BASICEDUC** (Basic Education):
    - Media: 98.64
    - Mínimo: 3.7, Máximo: 152.3
- **ESI** (Efficiency Subindex):
    - Media: 94.94
    - Mínimo: 40.1, Máximo: 157.4
- **HIGHEREDUC** (Higher Education):
    - Media: 97.83
    - Mínimo: 27.7, Máximo: 155.4

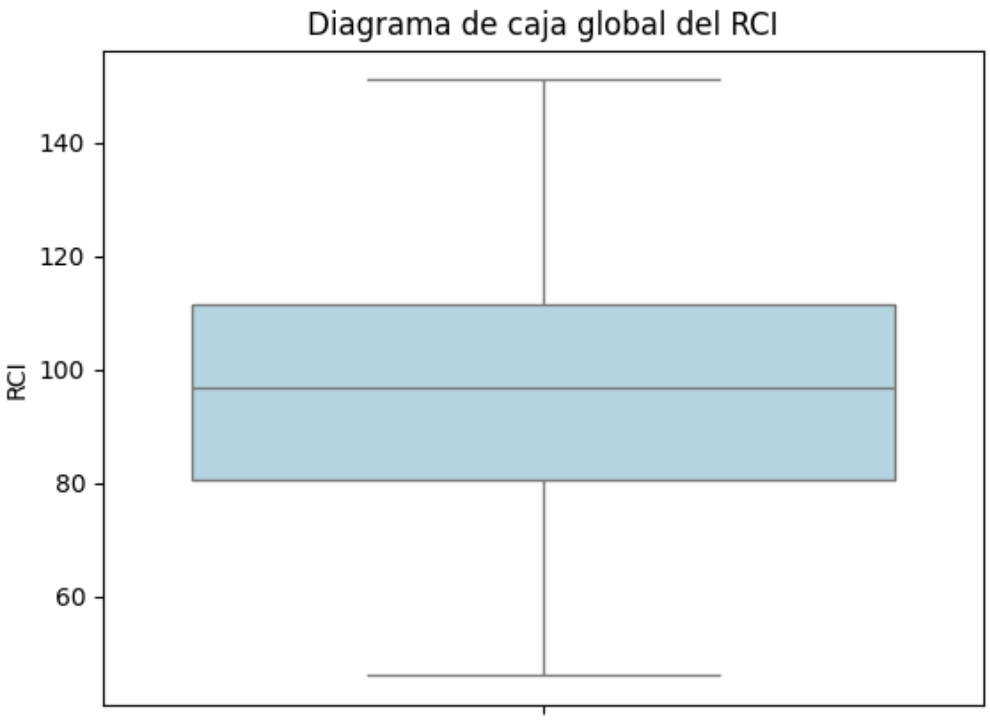
- **LABMARKET** (Labor Market):
  - Media: 99.05
  - Mínimo: 12.0, Máximo: 126.8
- **MARKETSIZE** (Market Size):
  - Media: 80.99
  - Mínimo: 3.9, Máximo: 274.6
- **ISI** (Innovation Subindex):
  - Media: 95.68
  - Mínimo: 23.7, Máximo: 166.9
- **TECHREAD** (Technological Readiness):
  - Media: 99.09
  - Mínimo: 13.1, Máximo: 175.7
- **BUSINESSSOPH** (Business Sophistication):
  - Media: 94.76
  - Mínimo: 12.5, Máximo: 184.4
- **INNOV** (Innovation):
  - Media: 92.16
  - Mínimo: 12.9, Máximo: 193.8
- **GDPPC** (GDP per capita):
  - Media: 93.41
  - Mínimo: 28.7, Máximo: 259.63

El código empleado es:

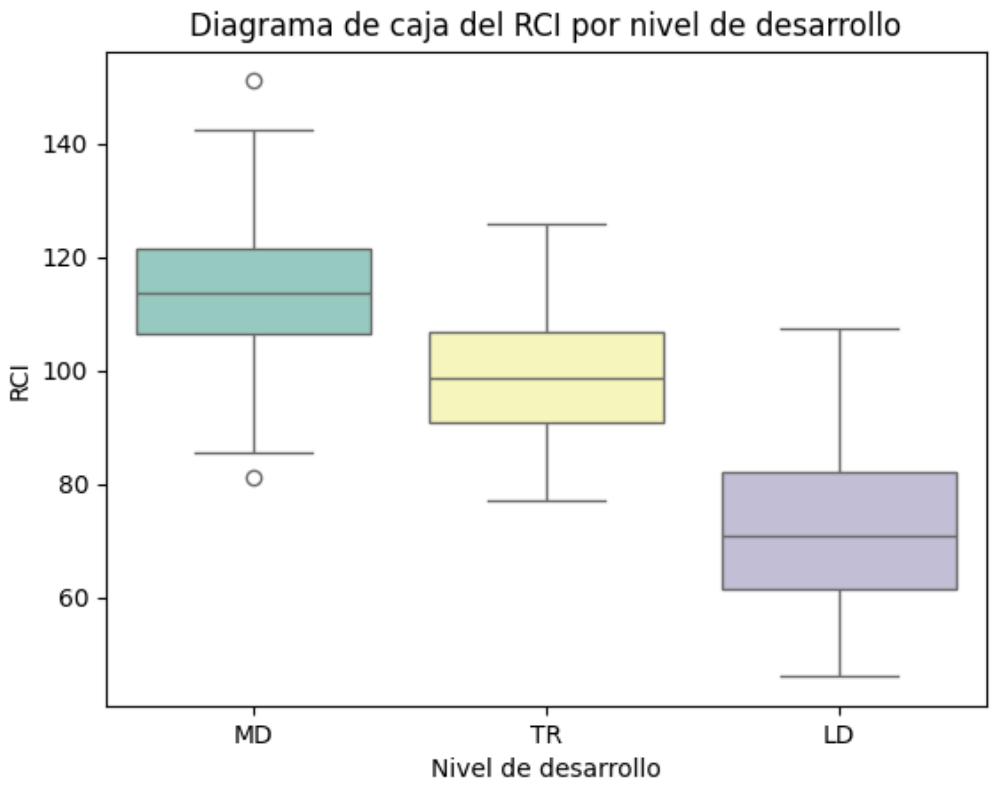
```
# Calcular el máximo y mínimo de cada columna numérica
max_values = data.max(numeric_only=True)
min_values = data.min(numeric_only=True)

# Mostrar los resultados
print("Valores Máximos:\n", max_values)
print("Valores Mínimos:\n", min_values)
```

3.

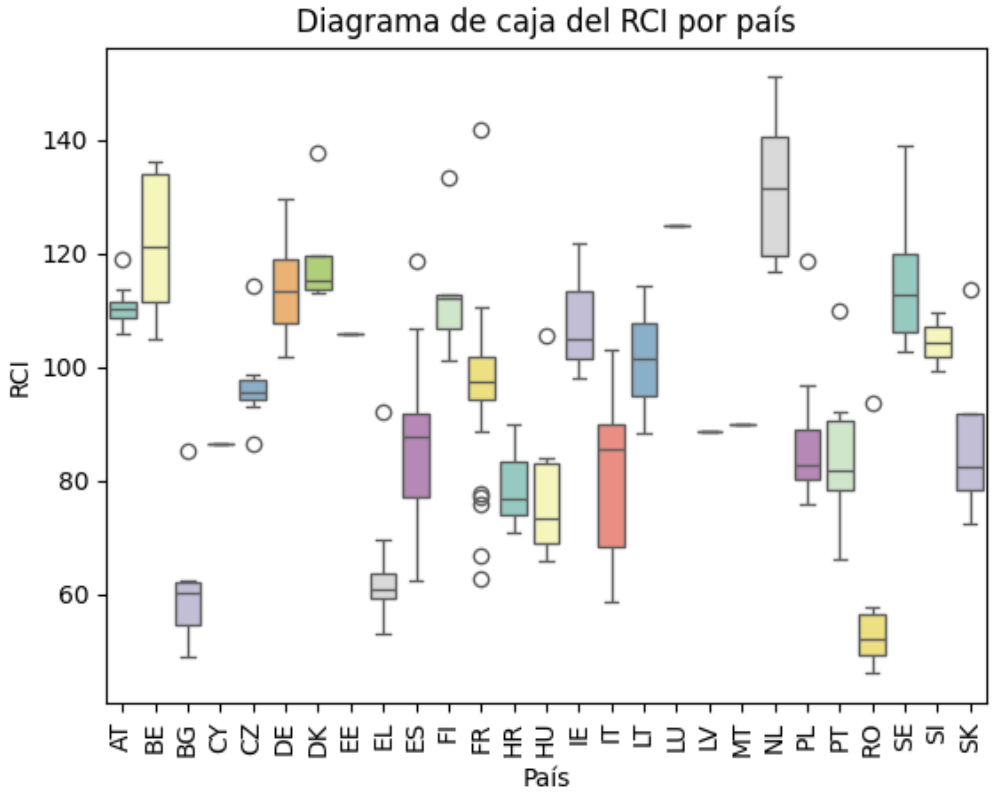
Se muestra los diagramas de caja del ICR generados tanto por RStudio:



A continuación, se muestran los diagramas de caja del ICR segmentados por región generados por RStudio:



A continuación, se muestran los diagramas de caja del ICR segmentados por países generados por RStudio:



Los respectivos códigos de RStudio desarrollados para llevar a cabo los diagramas son:

```
# Boxplot global del RCI
sns.boxplot(y='RCI', data=data, color="lightblue")
plt.title("Diagrama de caja global del RCI")
plt.ylabel("RCI")
plt.show()

# Boxplot del RCI condicionado por el nivel de desarrollo
sns.boxplot(x='DEVLEV', y='RCI', data=datos, palette="Set3")
plt.title("Diagrama de caja del RCI por nivel de desarrollo")
plt.xlabel("Nivel de desarrollo")
plt.ylabel("RCI")
plt.show()

# Boxplot del RCI condicionado por país
sns.boxplot(x='CNTR', y='RCI', data=datos, palette="Set3")
plt.title("Diagrama de caja del RCI por país")
plt.xlabel("País")
plt.ylabel("RCI")
plt.xticks(rotation=90)  # Rotar etiquetas de país
plt.show()
```

4.

10 mayores valores de ICR por país:

Código RStudio:

```
# Obtener los 10 mayores valores de RCI por país
top_10_RCI_pais = datos.groupby('CNTR').apply(lambda x: x.nlargest(10,
'RCI')).reset_index(drop=True)
print(top_10_RCI_pais)
```

Excel con los resultados:

top_10_RCI_pais.csv

10 mayores valores de ICR por región:

```
# Obtener los 10 mayores valores de RCI por región
top_10_RCI_region = datos.groupby('NUTS2ID').apply(lambda x:
x.nlargest(10, 'RCI')).reset_index(drop=True)
print(top_10_RCI_region)
```

Excel con los resultados:

top_10_RCI_region.csv

10 menores valores de ICR por país:

Código RStudio:

```
# Obtener los 10 menores valores de RCI por país
bottom_10_RCI_pais = datos.groupby('CNTR').apply(lambda x:
x.nsmallest(10, 'RCI')).reset_index(drop=True)
print(bottom_10_RCI_pais)
```

Excel con los resultados:

bottom_10_RCI_pais.csv

10 menores valores de ICR por región:

```
# Obtener los 10 menores valores de RCI por región
bottom_10_RCI_region = datos.groupby('NUTS2ID').apply(lambda x:
x.nsmallest(10, 'RCI')).reset_index(drop=True)
print(bottom_10_RCI_region)
```

Excel con los resultados:

bottom_10_RCI_region.csv

5.

El archivo es de tipo GeoJSON, este está estructurado en función de una serie de características:

Tiene una geometría MultiPolygon, es decir, las regiones están formadas por múltiples áreas discontinuas

Presenta unos atributos los cuales muestran el nombre de la región, el código NUTS (que identifica únicamente cada región) y una serie de coordenadas asociada a cada código

El código utilizado:

```
europe_map =
gpd.read_file("C:/Users/November/Downloads/EU_NUTS2.geojson")

print(europe_map.head())
print(europe_map.info())
```

6.

Para fusionar los ficheros EU_NUTS2.geojson y RCI_2_0_scores.xlsx utilizaremos el siguiente código de R:

```
# Fusionar los datos con el archivo GeoJSON
merged_data = europe_map.merge(datos, left_on="NUTS2ID",
right_on="NUTS2ID")
```

Una vez hemos juntado los dos grupos de datos procedemos a estudiar sus características:

Este nuevo conjunto de datos tiene una clase sf y data.frame, es decir, contiene datos geoespaciales y datos numéricos.

La geometría de cada observación es de tipo Multipolygon, es decir, cada región está formada por más de un lugar. Podemos encontrar una gran cantidad de variables en este conjunto de datos:

- **NUTS2ID**: Código de identificación de las regiones NUTS 2 (formato de texto). Ejemplos: "AL01", "AL02", "AT11", etc.
- **AGGCOUNT**: Número de regiones agregadas. La mayoría de los valores son 1, lo que indica que no hay agregación significativa.
- **REGION**: Nombre de la región, aunque muchos están vacíos (NA).

- **CNTR**: Código de país (formato de texto, por ejemplo, "AT" para Austria), pero hay varios NA.
- **DEVLEV**: Nivel de desarrollo regional, con valores como "TR" (transición) y algunos NA.
- **RANK**: Rango del Índice de Competitividad Regional (ICR) de las regiones. Tiene algunos valores NA.

Variables socioeconómicas: Incluyen diversos índices y puntajes sobre competitividad, infraestructura, educación, etc. Ejemplos:

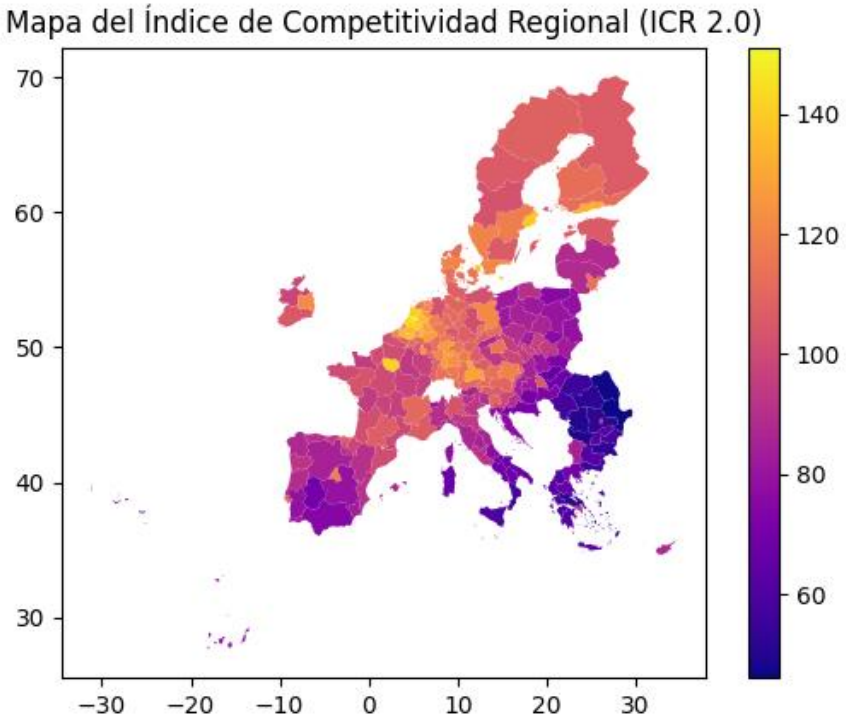
- **RCI**: Índice de Competitividad Regional (valor medio: 96.37).
- **GDPPC**: PIB per cápita (valor medio: 94.21).
- **INFRASTR**, **HEALTH**, **BASICEDUC**, **LABMARKET**, **INNOV**, entre otros.

El Código utilizado para obtener la información es el siguiente:

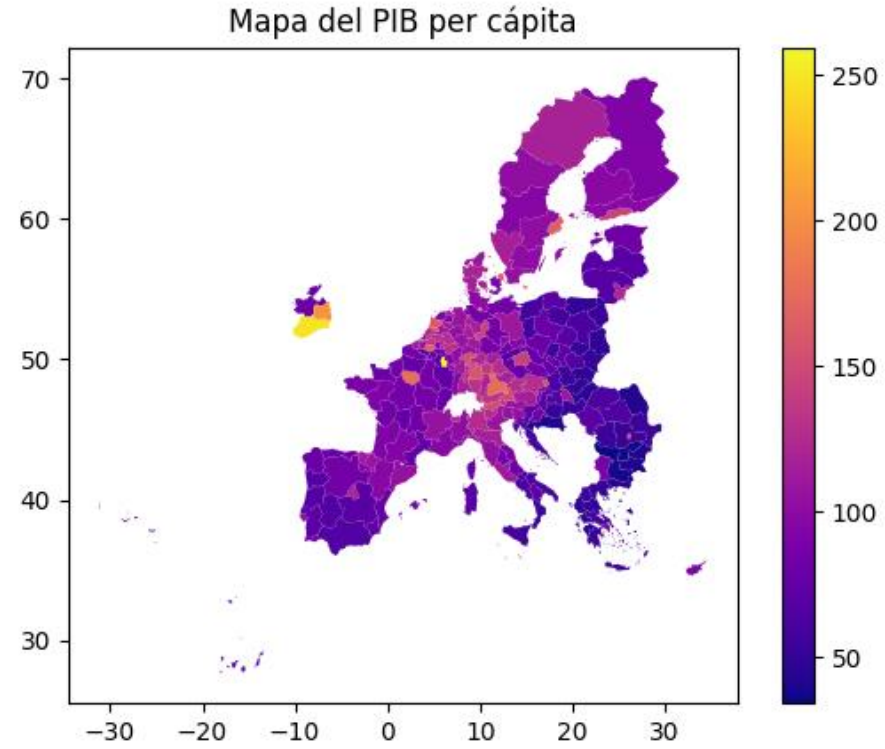
```
# Ver las primeras filas y estructura del nuevo dataset
print(merged_data.head())
print(merged_data.info())
```

7.

A continuación, se muestra un mapa en el que se exponen los niveles de competitividad regional en una escala de colores:



A continuación, se muestra un mapa en el que se expone el PIB per cápita por región a través de una escala de colores:



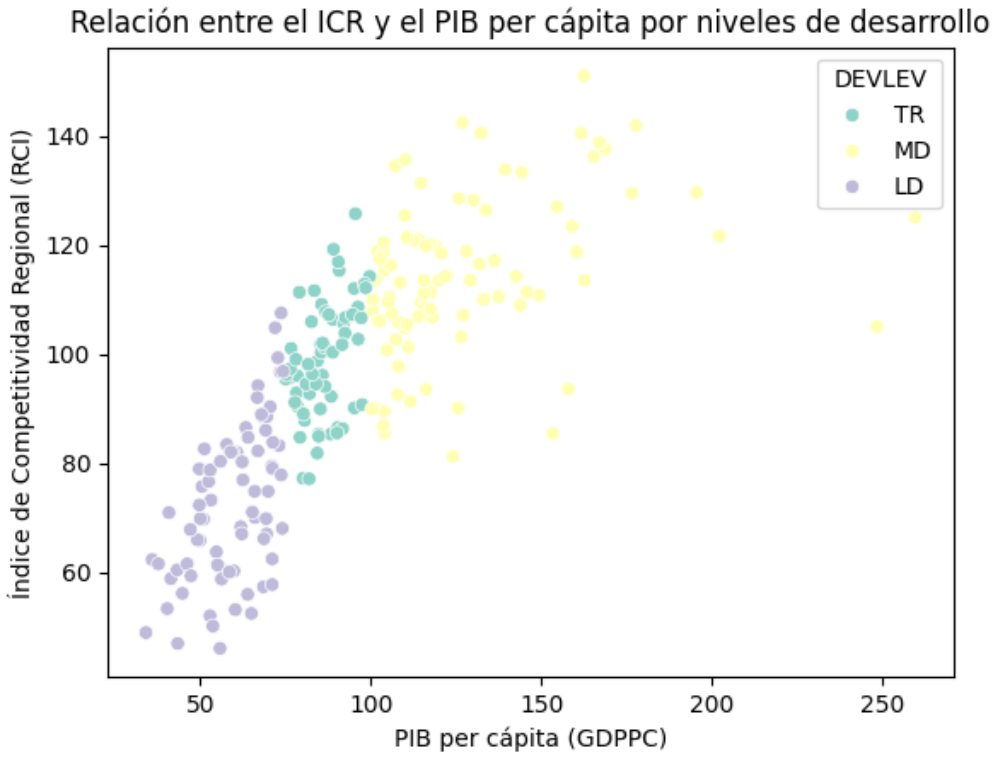
Tras observar ambos mapas podemos sospechar de la existencia de una relación entre el PIB per cápita y el índice de competitividad regional.

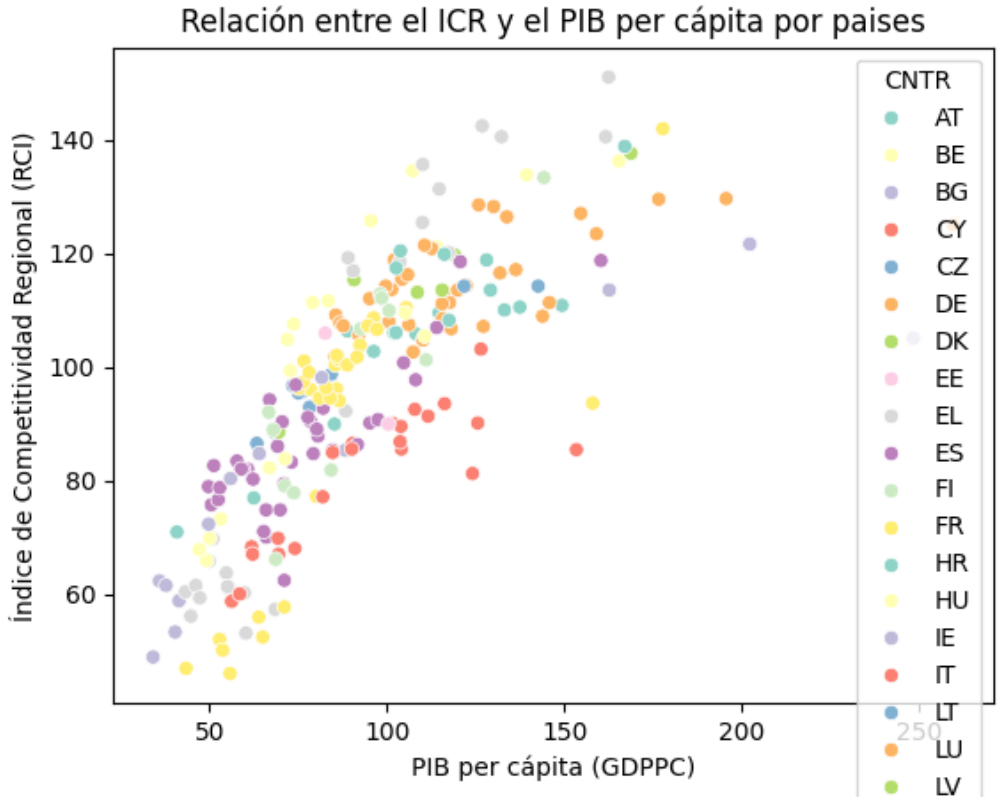
El Código empleado para realizar los respectivos mapas es:

```
# Mapa del Índice de Competitividad Regional (RCI 2.0)
merged_data.plot(column='RCI', cmap='plasma', legend=True)
plt.title("Mapa del Índice de Competitividad Regional (ICR 2.0)")
plt.show()

# Mapa del PIB per cápita (GDPPC)
merged_data.plot(column='GDPPC', cmap='plasma', legend=True)
plt.title("Mapa del PIB per cápita")
plt.show()
```

8.

En estas dos imágenes se muestra un diagrama de puntos que muestra la relación entre el ICR (eje y) y el PIB per cápita (eje x) segmentando tanto por país como por niveles de desarrollo regional:


Relación entre el ICR y el PIB per cápita por niveles de desarrollo
DEVLEV
TR
MD
LD
Índice de Competitividad Regional (RCI)
140
120
100
80
60
50
100
150
200
250
PIB per cápita (GDPPC)


Relación entre el ICR y el PIB per cápita por paises
CNTR
AT
BE
BG
CY
CZ
DE
DK
EE
EL
ES
FI
FR
HR
HU
IE
IT
LT
LU
LV
Índice de Competitividad Regional (RCI)
140
120
100
80
60
50
100
150
200
250
PIB per cápita (GDPPC)

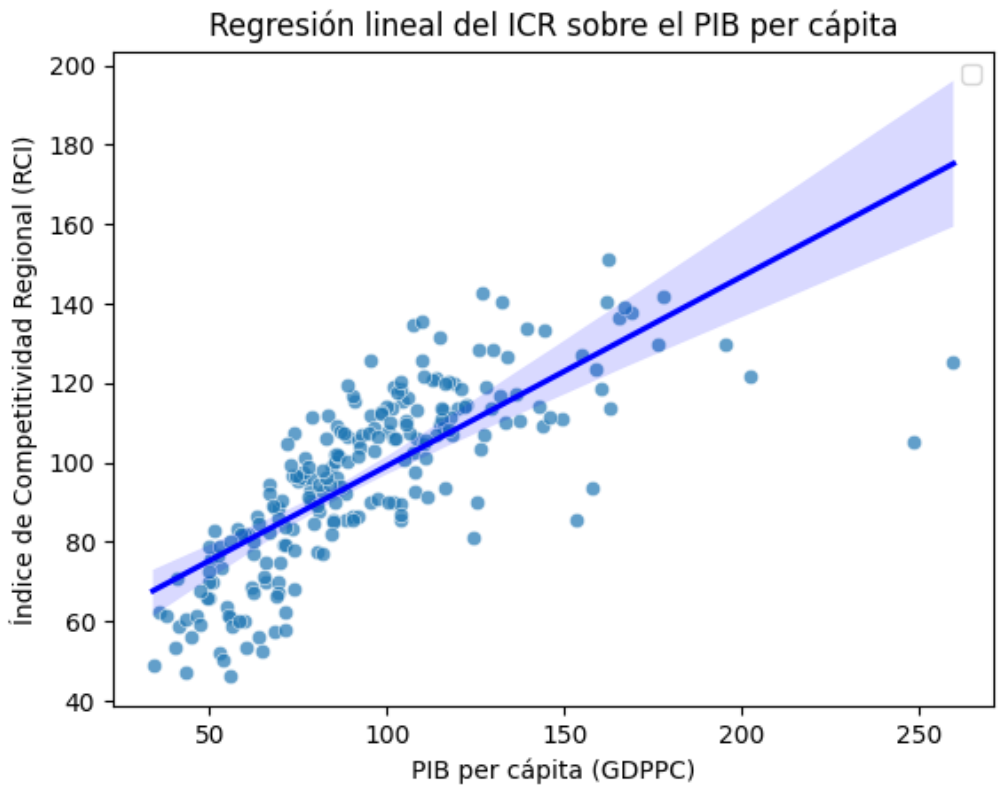
En este último gráfico podemos observar una relación directa entre el PIB per cápita y el ICR, esta relación tiene un carácter lineal creciente con una pendiente muy pronunciada en el caso de LD (Less developed regions) mientras que en las regiones TR (transition regions) se mantiene la relación lineal directa, con una pendiente más pronunciada. Por último, en las MD (Mode developed regions) no se observa una relación tan fuerte entre ambas variables, aunque si que es directa.

El código desarrollado para realizar los gráficos anteriores es:

```
# Gráfico de dispersión de RCI y GDPPC por paises
sns.scatterplot(x='GDPPC', y='RCI', hue='CNTR', data=merged_data,
palette='Set3')
plt.title("Relación entre el ICR y el PIB per cápita por paises")
plt.xlabel("PIB per cápita (GDPPC)")
plt.ylabel("Índice de Competitividad Regional (RCI)")
plt.show()


# Gráfico de dispersión de RCI y GDPPC por nivel de desarrollo
sns.scatterplot(x='GDPPC', y='RCI', hue='DEVLEV', data=merged_data,
palette='Set3')
plt.title("Relación entre el ICR y el PIB per cápita por niveles de
desarrollo")
plt.xlabel("PIB per cápita (GDPPC)")
plt.ylabel("Índice de Competitividad Regional (RCI)")
plt.show()
```

9.

A continuación, podemos observar un diagrama de puntos que permite estudiar la relación entre el ICR (eje y) y el PIB per cápita (eje x):



Esta recta de regresión presenta un R^2 de 0.5825 y un error residual de 14.17, lo que indica una relación directa de grado medio entre las variables ICR y PIB per cápita, es decir, cuanto mayor sea el PIB per cápita de una región mayore será su ICR y viceversa.

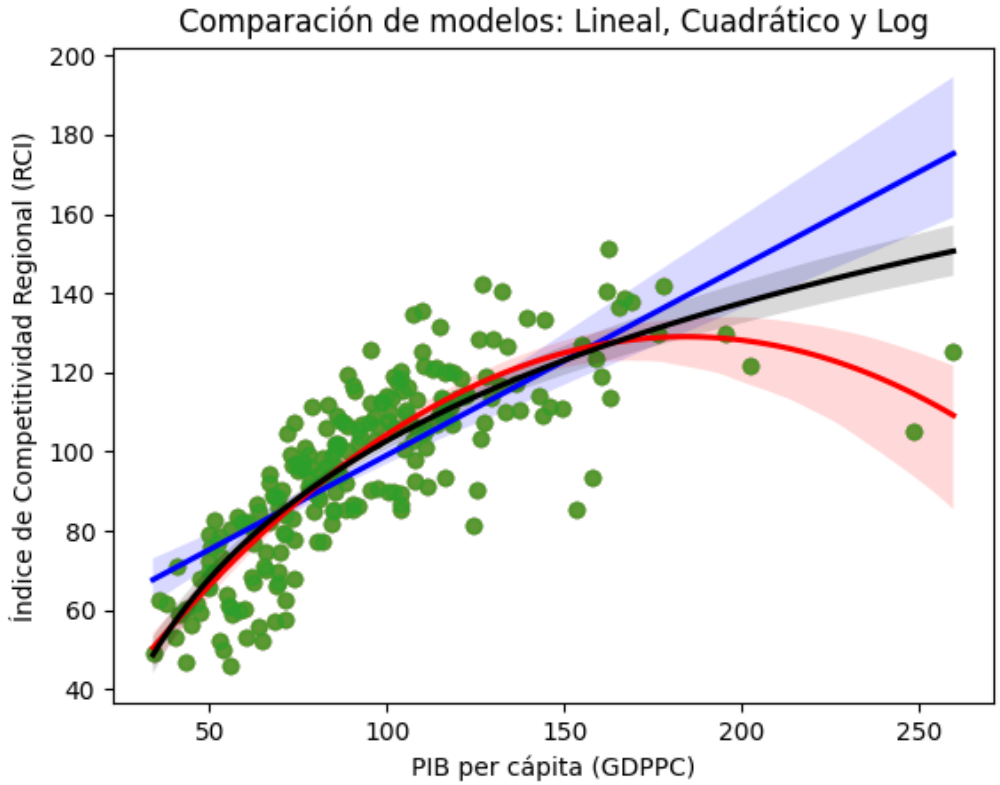
El código implementado para desarrollar el gráfico es el siguiente:

```
# Crear el modelo lineal de regresión
X = merged_data['GDPPC']
y = merged_data['RCI']
X = sm.add_constant(X)
modelo_regresion = sm.OLS(y, X).fit()
print(modelo_regresion.summary())

#Representación gráfica
sns.scatterplot(data=merged_data, x="GDPPC", y="RCI", alpha=0.7)
sns.regplot(data=merged_data, x="GDPPC", y="RCI", scatter=False,
color="blue", line_kws={"label":"Ajuste Lineal"})
plt.title("Regresión lineal del ICR sobre el PIB per cápita")
plt.xlabel("PIB per cápita (GDPPC)")
plt.ylabel("Índice de Competitividad Regional (RCI)")
plt.legend()
```

```
plt.show()
```

10.

Una comparación gráfica de los 3 modelos:



La función azul representa la recta de regresión lineal, la de color rojo la de la regresión cuadrática y la negra la regresión logarítmica.

En el caso de la regresión cuadrática obtenemos un R^2 de 0.7121 y un error residual de 12.05, es decir, este modelo se ajusta mejor a los datos que el modelo lineal.

Por otra parte, el modelo logarítmico presenta un R^2 de 0.6835 y un error residual de 12.6, es decir, es ligeramente peor que el modelo anterior.

Estudiando los AIC de los 3 modelos observamos lo siguiente:

```
##                   df       AIC
## modelo_regresion   3 1877.8158
## modelo_cuadratico  4 1794.6888
## modelo_log         3 1814.4125
```

Como sospechábamos el modelo con menor AIC es el modelo cuadrático, por tant siguiendo el criterio de AIC el modelo que mejor se ajusta a los datos es el modelo cuadrático

El código implementado para desarrollar el gráfico es el siguiente:

```
# Crear el modelo lineal de regresión
X = merged_data['GDPPC']
y = merged_data['RCI']
X = sm.add_constant(X)
modelo_regresion = sm.OLS(y, X).fit()
print(modelo_regresion.summary())

# Modelo cuadrático
X_cuadratico = sm.add_constant(np.column_stack((merged_data['GDPPC'],
merged_data['GDPPC']**2)))
modelo_cuadratico = sm.OLS(y, X_cuadratico).fit()
print(modelo_cuadratico.summary())

# Modelo logarítmico
X_log = sm.add_constant(np.log(merged_data['GDPPC']))
modelo_loglog = sm.OLS(y, X_log).fit()
print(modelo_loglog.summary())

# Comparar AIC entre los modelos
print("AIC comparativo:", modelo_regresion.aic, modelo_cuadratico.aic,
modelo_loglog.aic)

# Comparación de modelos en gráfico
sns.regplot(x='GDPPC', y='RCI', data=merged_data,
scatter_kws={'alpha':0.7}, line_kws={'color':'blue'})
sns.regplot(x='GDPPC', y='RCI', data=merged_data,
scatter_kws={'alpha':0.7}, line_kws={'color':'red'}, order=2)
sns.regplot(x=np.log(merged_data['GDPPC']), y='RCI', data=merged_data,
scatter_kws={'alpha':0.7}, line_kws={'color':'green'},
x_estimator=np.log)
plt.title("Comparación de modelos: Lineal, Cuadrático y Log")
plt.xlabel("PIB per cápita (GDPPC)")
plt.ylabel("Índice de Competitividad Regional (RCI)")
plt.show()
```