Measuring Democracy Challenges of Individual and General Measures

April 7, 2024

by Pedro Huet López

[1]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
import statsmodels.api as sm
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

file_path = r"C:\Users\pedro\Downloads\Data excercise\Untitled␣
↪Folder\ESS10\ESS10.csv"
euro_c_data = pd.read_csv(file_path, low_memory=False)

sum_euro_c_data = euro_c_data
```

[2]:
```
#Preparing data

sum_euro_c_data['vote'].replace(2, 0, inplace=True)
sum_euro_c_data = sum_euro_c_data.loc[sum_euro_c_data['vote'].isin([0, 1])]

sum_euro_c_data['dscrgrp'].replace(2, 0, inplace=True)
sum_euro_c_data = sum_euro_c_data.loc[sum_euro_c_data['dscrgrp'].isin([0, 1])]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['lrnobed'].isin([1, 2, 3, 4,␣
↪5])]

recode_map = {0: 10, 1: 9, 2: 8, 3: 7, 4: 6, 5: 5, 6: 4, 7: 3, 8: 2, 9: 1, 10:␣
↪0}
sum_euro_c_data['accalaw'] = sum_euro_c_data['accalaw'].map(recode_map)
sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['accalaw'].isin(range(11))]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['implvdm'].isin([0, 1, 2, 3,␣
↪4, 5, 6, 7, 8, 9, 10])]
```

```
sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['loylead'].isin([1, 2, 3, 4␣
 ↪,5])]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['secgrdec'].isin([1, 2, 3, 4␣
 ↪,5])]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['hmsacld'].isin([1, 2, 3, 4␣
 ↪,5])]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['chpldmi'].isin([0, 1, 2, 3,␣
 ↪4, 5, 6, 7, 8, 9, 10])]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['trstep'].isin([0, 1, 2, 3,␣
 ↪4, 5, 6, 7, 8, 9, 10])]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['hinctnta'].isin([0, 1, 2, 3,␣
 ↪4, 5, 6, 7, 8, 9, 10])]

sum_euro_c_data = sum_euro_c_data[sum_euro_c_data['rlgdgr'].isin([0, 1, 2, 3,␣
 ↪4, 5, 6, 7, 8, 9, 10])]

print("Number of unique observations for 'cntry':", sum_euro_c_data['cntry'].
 ↪nunique())

import numpy as np

conditions = [
    sum_euro_c_data['cntry'] == 'MK',
    sum_euro_c_data['cntry'] == 'NO',
    sum_euro_c_data['cntry'] == 'IS',
    sum_euro_c_data['cntry'] == 'FI',
    sum_euro_c_data['cntry'] == 'CH',
    sum_euro_c_data['cntry'] == 'GR',
    sum_euro_c_data['cntry'] == 'NL',
    sum_euro_c_data['cntry'] == 'GB',
    sum_euro_c_data['cntry'] == 'IT',
    sum_euro_c_data['cntry'] == 'PT',
    sum_euro_c_data['cntry'] == 'IE',
    sum_euro_c_data['cntry'] == 'EE',
    sum_euro_c_data['cntry'] == 'BG',
    sum_euro_c_data['cntry'] == 'HU',
    sum_euro_c_data['cntry'] == 'BE',
    sum_euro_c_data['cntry'] == 'HR',
    sum_euro_c_data['cntry'] == 'SI',
    sum_euro_c_data['cntry'] == 'SK',
    sum_euro_c_data['cntry'] == 'CZ',
```

```
    sum_euro_c_data['cntry'] == 'LT'
]

values = [
    5.89,
    9.81,
    9.37,
    9.20,
    8.83,
    7.39,
    8.96,
    8.54,
    7.74,
    7.90,
    9.05,
    7.84,
    6.71,
    6.56,
    7.51,
    6.50,
    7.54,
    6.97,
    7.67,
    7.13
]

sum_euro_c_data['e_dem_index'] = np.select(conditions, values, default=np.nan)

print(sum_euro_c_data)

cntry_value_counts = sum_euro_c_data['cntry'].value_counts()
print("Number of observations with different values in 'cntry':")
```

```
C:\Users\pedro\AppData\Local\Temp\ipykernel_8780\3860888525.py:6:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  sum_euro_c_data['dscrgrp'].replace(2, 0, inplace=True)
```

```
Number of unique observations for 'cntry': 20
             name  essround  edition    proddate   idno cntry   dweight  \
7      ESS10e03_2        10      3.2  02.11.2023  10088    BE  0.898875
8      ESS10e03_2        10      3.2  02.11.2023  10089    BE  1.087741
9      ESS10e03_2        10      3.2  02.11.2023  10104    BE  1.079369
12     ESS10e03_2        10      3.2  02.11.2023  10133    BE  1.079369
17     ESS10e03_2        10      3.2  02.11.2023  10220    BE  1.030007
```

```
…           …       …       …          …    …   …       …
37595  ESS10e03_2        10      3.2  02.11.2023  27677    SK  1.069332
37599  ESS10e03_2        10      3.2  02.11.2023  27746    SK  0.516693
37603  ESS10e03_2        10      3.2  02.11.2023  27785    SK  0.578790
37604  ESS10e03_2        10      3.2  02.11.2023  27787    SK  1.944061
37606  ESS10e03_2        10      3.2  02.11.2023  27808    SK  0.515714

        pspwght   pweight  anweight  …                inwde  \
7      0.671588  0.718075  0.482251  …  2022-04-22 17:04:00
8      1.239038  0.718075  0.889723  …  2022-04-04 11:34:00
9      2.133335  0.718075  1.531895  …  2022-06-15 11:35:00
12     1.939416  0.718075  1.392647  …  2022-05-25 19:17:00
17     0.704808  0.718075  0.506105  …  2022-08-19 15:55:00
…         …       …       …  …                   …
37595  0.863272  0.323800  0.279528  …  2021-07-27 13:12:58
37599  0.418275  0.323800  0.135437  …  2021-08-09 17:19:06
37603  0.467257  0.323800  0.151298  …  2021-08-04 16:03:09
37604  1.313235  0.323800  0.425226  …  2021-05-26 14:48:51
37606  0.339385  0.323800  0.109893  …  2021-06-08 14:30:41

                     jinws                jinwe  inwtm  mode  domain  \
7      2022-04-22 17:02:00  2022-04-22 17:04:00   94.0     1     1.0
8      2022-04-04 11:32:00  2022-04-04 11:34:00   70.0     1     2.0
9      2022-06-15 11:34:00  2022-06-15 11:35:00   69.0     1     2.0
12     2022-05-25 19:17:00  2022-05-25 19:17:00   42.0     1     2.0
17     2022-08-19 15:53:00  2022-08-19 15:55:00   64.0     1     2.0
…                      …                    …    …   …      …
37595  2021-07-27 13:11:45  2021-07-27 13:17:40   56.0     1     1.0
37599  2021-08-09 17:18:26  2021-08-09 17:20:13   49.0     1     1.0
37603  2021-08-04 16:01:52  2021-08-04 16:03:23   51.0     1     1.0
37604  2021-05-26 14:48:09  2021-05-26 14:49:04   39.0     1     1.0
37606  2021-06-08 14:29:01  2021-06-08 14:31:44   70.0     1     1.0

           prob  stratum    psu  e_dem_index
7      0.000389      147   2614         7.51
8      0.000322      198   2023         7.51
9      0.000324      197   2575         7.51
12     0.000324      197   2662         7.51
17     0.000340      200   2673         7.51
…         …       …     …           …
37595  0.000734     2620  27010         6.97
37599  0.001519     2635  27177         6.97
37603  0.001356     2623  27034         6.97
37604  0.000404     2617  26982         6.97
37606  0.001522     2610  27206         6.97

[13889 rows x 619 columns]
Number of observations with different values in 'cntry':
```

Question 1:

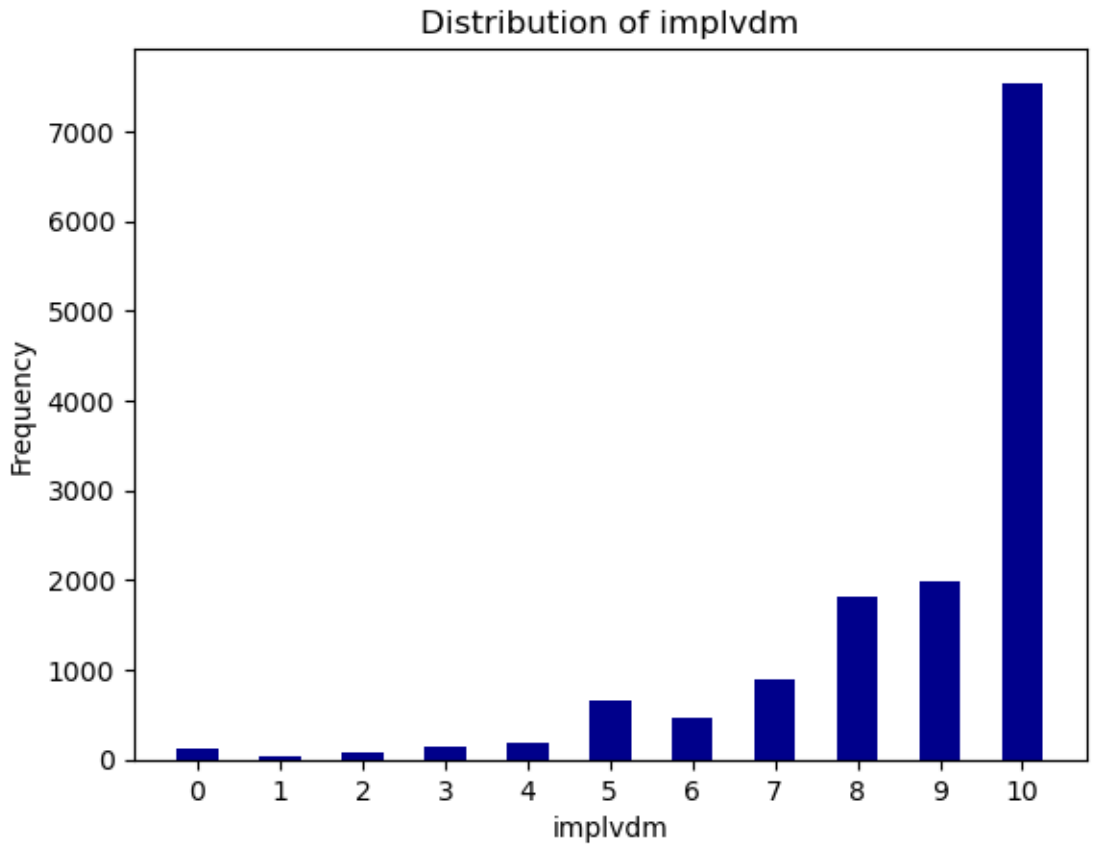
*1. If we were to profile individuals according to their democratic values, which countries display a higher support for democracy?

Tip: Choose a set of variables that capture the democratic values of individuals and reduce their dimensionality to a single "support-for-democracy" measure (be creative).*

A good first approach to observe the democratic values of individuals would be to find the variable in the dataframe that most closely portrays this question. When looking into the data, we find that "implvdm" seems to be the closest to fulfill the criteria: "How important is it for you to live in democratically governed country".

Although the question seems to straight forward and directly measuring what we are interested in, we run into a problem: social desirability bias. Researchers (Edwards, 1957; Krumpal, 2013) have found that individuals tend to select categories that don't personally reflect their feelings towards a subject because they think their answer will be judged by the researcher; we could think that one such case is the sentiment people hold towards democracy. When viewing the distribution of the implvdm variable, we find that it is massively skewed towards a positive view on democracy:

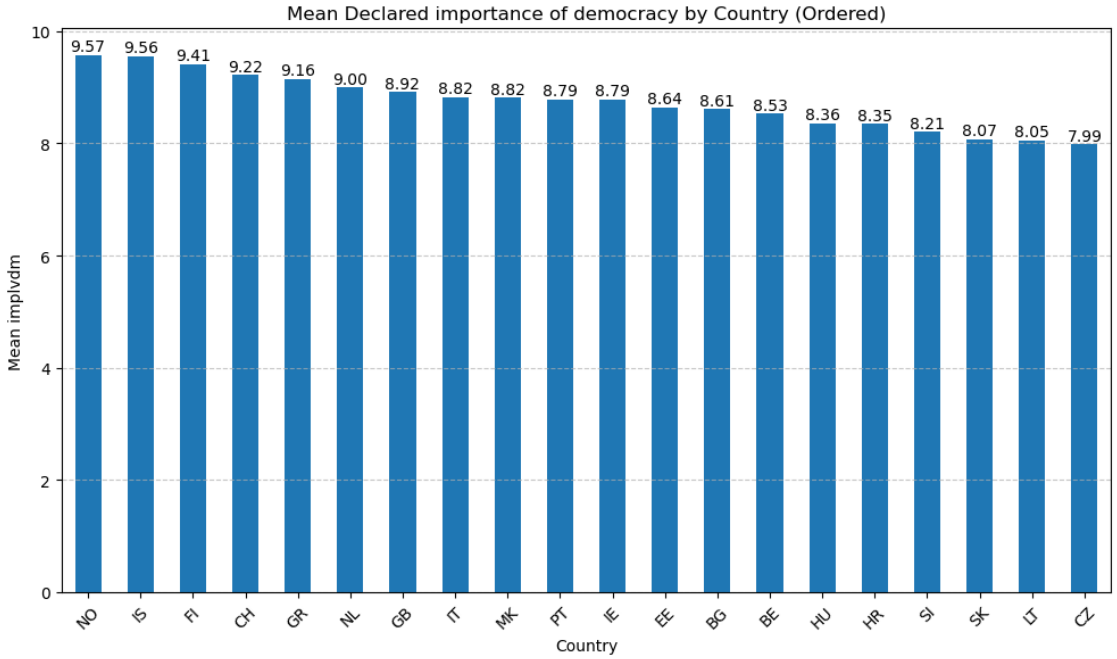
[3]:
```
plt.hist(sum_euro_c_data['implvdm'], bins=[-0.5, 0.5, 1.5, 2.5, 3.5, 4.5, 5.5,␣
↪6.5, 7.5, 8.5, 9.5, 10.5], rwidth=0.5, color='darkblue')

plt.xlabel('implvdm')
plt.ylabel('Frequency')
plt.title('Distribution of implvdm')

plt.xticks(range(0, 11))

plt.show()
```



As a first approach, we use this conventional measure of importance of democracy to see how the mean distribution of people's stance towards democracy by country looks like:

[4]:
```
#Conventional measure
mean_implvdm_by_country = sum_euro_c_data.groupby('cntry')['implvdm'].mean()
mean_implvdm_by_country_sorted = mean_implvdm_by_country.
 ↪sort_values(ascending=False)

plt.figure(figsize=(10, 6))
mean_implvdm_by_country_sorted.plot(kind='bar')
plt.title('Mean Declared importance of democracy by Country (Ordered)')
plt.xlabel('Country')
plt.ylabel('Mean implvdm')
plt.xticks(rotation=45)

for i, mean_value in enumerate(mean_implvdm_by_country_sorted):
    plt.text(i, mean_value + 0.01, f'{mean_value:.2f}', ha='center',␣
 ↪va='bottom')

plt.grid(axis='y', linestyle='--', alpha=0.7)
```

```
plt.tight_layout()
plt.show()
```

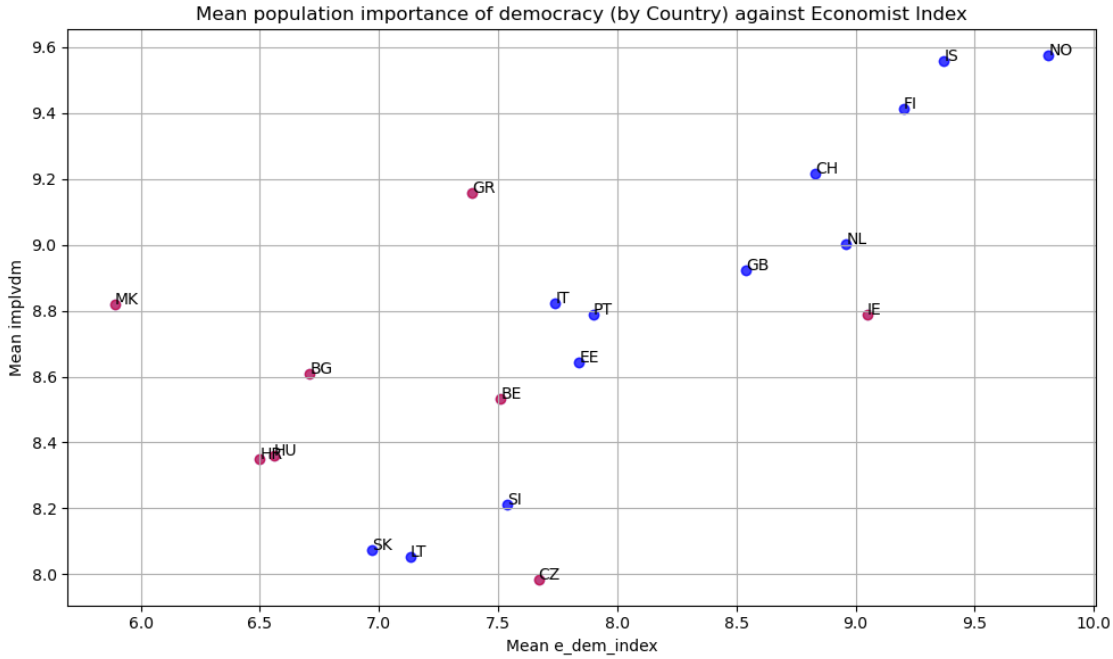


This distribution would seem fine, yet a good way to verify its accuracy would be to compare this result to a more standard measure of democratic values, as the Economist's Democracy Index. When comparing both, we obtain the following plot:

[5]:
```
#Comparison to the Economist Index

mean_values_by_country = sum_euro_c_data.groupby('cntry').mean()

z_scores = (mean_values_by_country[['implvdm', 'e_dem_index']] -␣
 ↪mean_values_by_country[['implvdm', 'e_dem_index']].mean()) /␣
 ↪mean_values_by_country[['implvdm', 'e_dem_index']].std()

abs_z_scores = np.abs(z_scores)

countries_to_color_red = ['MK', 'GR', 'BE', 'HU', 'HR', 'IE', 'CZ','BG']

plt.figure(figsize=(10, 6))
plt.scatter(mean_values_by_country['e_dem_index'],␣
 ↪mean_values_by_country['implvdm'], color='blue', alpha=0.5)

for country in mean_values_by_country.index:
    color = 'red' if country in countries_to_color_red else 'blue'
```

```
    plt.scatter(mean_values_by_country.loc[country, 'e_dem_index'],␣
 ↪mean_values_by_country.loc[country, 'implvdm'], color=color, alpha=0.5)
    plt.text(mean_values_by_country.loc[country, 'e_dem_index'],␣
 ↪mean_values_by_country.loc[country, 'implvdm'], country)

plt.title('Mean population importance of democracy (by Country) against␣
 ↪Economist Index')
plt.xlabel('Mean e_dem_index')
plt.ylabel('Mean implvdm')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
C:\Users\pedro\AppData\Local\Temp\ipykernel_8780\2935930688.py:3: FutureWarning:
The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a
future version, numeric_only will default to False. Either specify numeric_only
or select only columns which should be valid for the function.
  mean_values_by_country = sum_euro_c_data.groupby('cntry').mean()
```



The results are mixed. On one, hand most if the top performing countries (Norway, Iceland, Finland, Switzerland, i.e.), as well as a few low performing countries (Slovenia, Slovakia, Lithuania) in the ESS10 survey's measure are located close to their Economist democratic index value counterpart. On the other hand, we see also some noticeable mismatches: while Greece (GR) and North Macedonia's (MK) national averages are placed very high in the survey's democracy metric, we observe that their democracy index from the Economist is much lower. On the other hand, other countries, like Ireland (IE) and Czechia (CZ) are ranked comparatively much lower in the ESS10

survey than in their position at the Economist index, showing more problems in accuracy. As expected, these results seem to show that using direct questions to rate individuals' democratic values usually falls into pitfalls involving individuals' disposition of giving strategic answers.

As a solution, I propose the creation of an index, composed out of a series of instrumental variables that are highly correlated to a person's disposition towards democracy, yet should not introduce important biases to our estimations.

A first variable that could be used to generate this index is if an individual tends to vote in his/her election. Using the survey data we find that this question is asked as "Voted in last national election", and is asked in a dichotomous manner (we only concentrate on the coded answers "Yes" and "No"). Since the basic definition of a democracy is that is a system in which decisions are made by the people (Aristotle [350 BCE] 2009), it would make sense to include this measure in the index. Although we might see a high number of voters turning out as a baseline, it could be a good differentiator for individuals not as passionate for democracy.

A second variable for the index could be if individuals have less contestation towards politicians and authorities. Since democracy tends to imply that voters are the decision makers in a political system, it is reasonable to expect individuals who are most obedient and submissive to authority might be less inclined to favor a system that proposes the opposite behavior. One of the ways best found by scholars (Feldman and Stenner, 1997) is to use the instrumental variable of children's education: parents who tend to value submission to an authority also tend to instruct their children to prioritize obeying authority. The question is measured as: "Obedience and respect for authority as the most important virtues children should learn."

Another way to measure this almost blind obedience to authority, concerning for democratic values, is seeing how important it is to individuals having loyalty to a leader. This is also measured directly in the survey via the question: "Country needs most loyalty towards its leaders".

Another useful variable to measure democratic values is to see respondent's perception on leaders that violate the law. Since modern representative democracy's foundations are based on limiting the power of elected authorities towards institutional boundaries in order to guarantee that they implement the will of the people (Madison et al,[1788] 2005), it would make sense that proponents of democracy would prefer leaders not violate legal boundaries. To measure this, in the survey there is the question of: "Acceptable for the country to have a strong leader above the law".

Another important measure is the capacity of a government of being responsive to the population. Since democracy is a system in which elected authorities should be held accountable by the population (Bourke,[1774] 1777; Bingham Powell, 2004), it would make sense that people who are democratic tend to prefer their authorities change their actions based on the will of the people. In order to measure this, the survey asks the question: "Important for democracy: government changes policies in response to what most people think".

Finally, a more recent breakthrough in the literature (Pantazi and Prooijen, 2023) has been the finding that individuals who tend to be less democratic also tend to be those that believe in political conspiracy theories. Since most of the creators of conspiracy theories tend to be populist politicians who want to stay in power and create these to try to justify their mistakes/failures to the population, the individuals who believe in conspiracy theories tend to be supporters of these antidemocratic politicians and their governments. In order to measure this, the survey asks: "[There is a] [s]mall secret group of people responsible for making all major decisions in world politics."

Normalizing and assigning the same weight to these 6 instrumental variables, we create an index.

When seeing its mean ranking per country, we get the following:

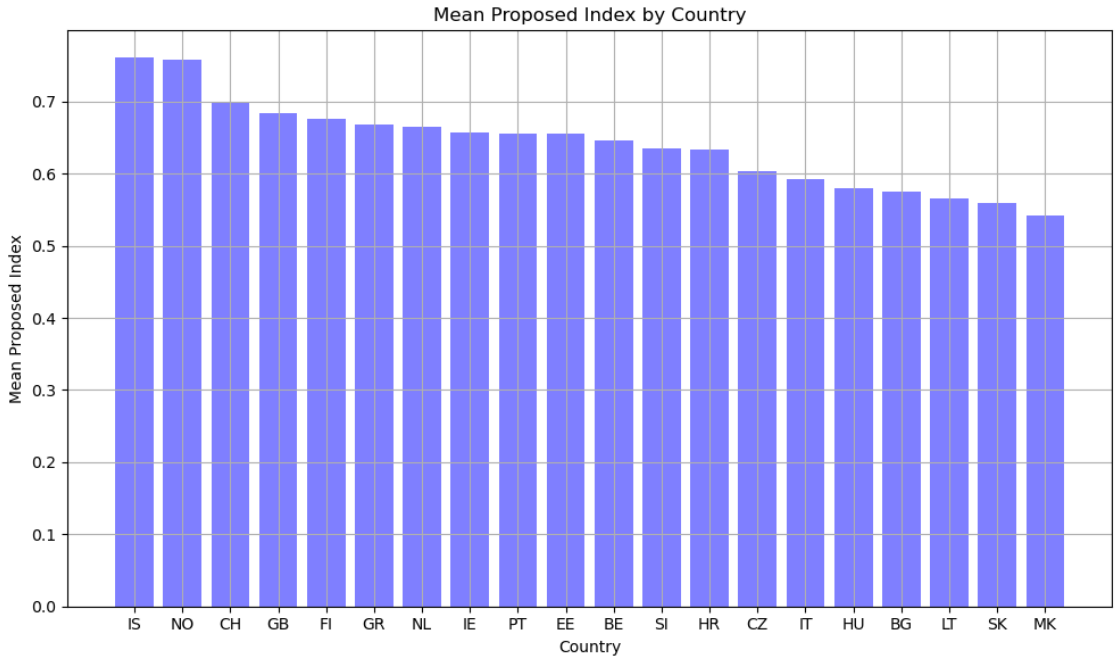
[6]:
```
#Proposed Index

from sklearn.preprocessing import MinMaxScaler

variables = ['vote', 'lrnobed', 'accalaw', 'chpldmi', 'loylead', 'secgrdec']

scaled_values = sum_euro_c_data[variables] / len(variables)
scaler = MinMaxScaler()
scaled_index = scaler.fit_transform(scaled_values.sum(axis=1).values.
 ↪reshape(-1, 1))

sum_euro_c_data['index'] = scaled_index
print(sum_euro_c_data)
```

```
            name  essround  edition    proddate   idno cntry   dweight  \
7      ESS10e03_2        10      3.2  02.11.2023  10088    BE  0.898875
8      ESS10e03_2        10      3.2  02.11.2023  10089    BE  1.087741
9      ESS10e03_2        10      3.2  02.11.2023  10104    BE  1.079369
12     ESS10e03_2        10      3.2  02.11.2023  10133    BE  1.079369
17     ESS10e03_2        10      3.2  02.11.2023  10220    BE  1.030007
…           …       …       …          …    …   …       …
37595  ESS10e03_2        10      3.2  02.11.2023  27677    SK  1.069332
37599  ESS10e03_2        10      3.2  02.11.2023  27746    SK  0.516693
37603  ESS10e03_2        10      3.2  02.11.2023  27785    SK  0.578790
37604  ESS10e03_2        10      3.2  02.11.2023  27787    SK  1.944061
37606  ESS10e03_2        10      3.2  02.11.2023  27808    SK  0.515714

        pspwght   pweight  anweight  …                jinws  \
7      0.671588  0.718075  0.482251  …  2022-04-22 17:02:00
8      1.239038  0.718075  0.889723  …  2022-04-04 11:32:00
9      2.133335  0.718075  1.531895  …  2022-06-15 11:34:00
12     1.939416  0.718075  1.392647  …  2022-05-25 19:17:00
17     0.704808  0.718075  0.506105  …  2022-08-19 15:53:00
…         …       …       …  …                   …
37595  0.863272  0.323800  0.279528  …  2021-07-27 13:11:45
37599  0.418275  0.323800  0.135437  …  2021-08-09 17:18:26
37603  0.467257  0.323800  0.151298  …  2021-08-04 16:01:52
37604  1.313235  0.323800  0.425226  …  2021-05-26 14:48:09
37606  0.339385  0.323800  0.109893  …  2021-06-08 14:29:01

                    jinwe  inwtm  mode  domain      prob  stratum    psu  \
7      2022-04-22 17:04:00   94.0     1     1.0  0.000389      147   2614
8      2022-04-04 11:34:00   70.0     1     2.0  0.000322      198   2023
9      2022-06-15 11:35:00   69.0     1     2.0  0.000324      197   2575
12     2022-05-25 19:17:00   42.0     1     2.0  0.000324      197   2662
17     2022-08-19 15:55:00   64.0     1     2.0  0.000340      200   2673
```

```
…                    …    …   …      …        …       …    …
37595  2021-07-27 13:17:40   56.0     1      1.0  0.000734     2620  27010
37599  2021-08-09 17:20:13   49.0     1      1.0  0.001519     2635  27177
37603  2021-08-04 16:03:23   51.0     1      1.0  0.001356     2623  27034
37604  2021-05-26 14:49:04   39.0     1      1.0  0.000404     2617  26982
37606  2021-06-08 14:31:44   70.0     1      1.0  0.001522     2610  27206

       e_dem_index    index
7             7.51  0.50000
8             7.51  0.71875
9             7.51  0.40625
12            7.51  0.53125
17            7.51  0.50000
…              …       …
37595         6.97  0.81250
37599         6.97  0.62500
37603         6.97  0.37500
37604         6.97  0.50000
37606         6.97  0.34375

[13889 rows x 620 columns]
```

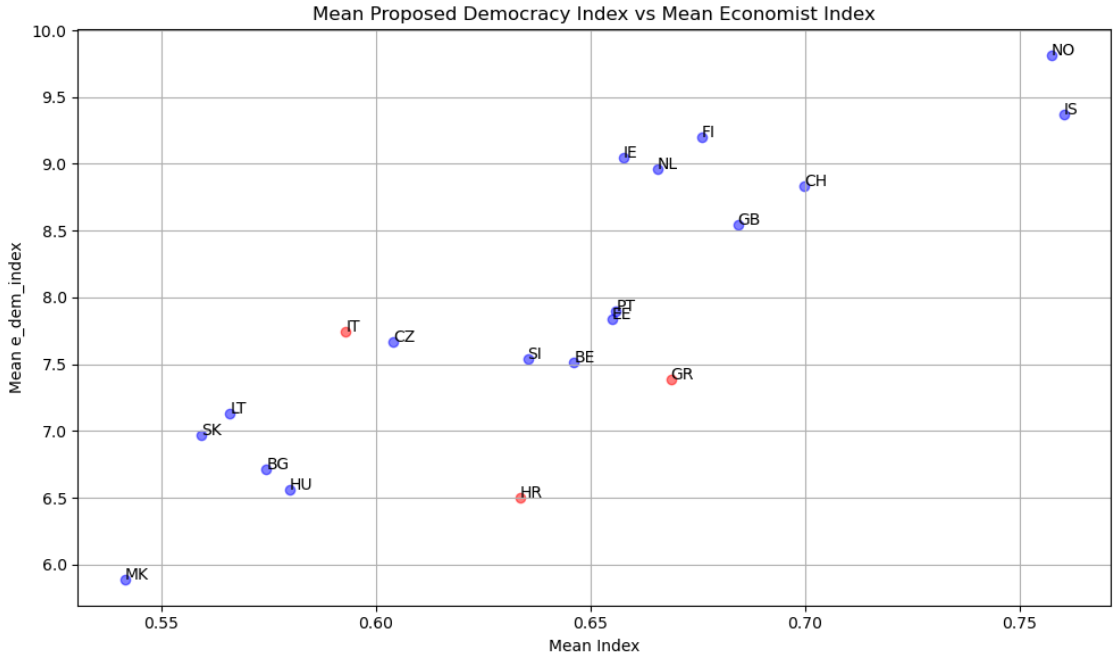
[7]:
```
mean_values_by_country = sum_euro_c_data.groupby('cntry').mean()

mean_values_by_country_sorted = mean_values_by_country.sort_values(by='index',␣
 ↪ascending=False)

countries_to_color_red = []

plt.figure(figsize=(10, 6))
for country, mean_index in mean_values_by_country_sorted.iterrows():
    color = 'red' if country in countries_to_color_red else 'blue'
    plt.bar(country, mean_index['index'], color=color, alpha=0.5)

plt.title('Mean Proposed Index by Country')
plt.xlabel('Country')
plt.ylabel('Mean Proposed Index')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
C:\Users\pedro\AppData\Local\Temp\ipykernel_8780\2454231227.py:1: FutureWarning:
The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a
future version, numeric_only will default to False. Either specify numeric_only
or select only columns which should be valid for the function.
  mean_values_by_country = sum_euro_c_data.groupby('cntry').mean()
```



[8]:
```
mean_values_by_country = sum_euro_c_data.groupby('cntry').mean()

countries_to_color_red = ['GR','HR','IT']

mean_values_by_country_sorted = mean_values_by_country.sort_values(by='index',␣
 ↪ascending=False)

plt.figure(figsize=(10, 6))
for country, mean_index in mean_values_by_country_sorted.iterrows():
    color = 'red' if country in countries_to_color_red else 'blue'
    plt.scatter(mean_index['index'], mean_index['e_dem_index'], color=color,␣
 ↪alpha=0.5)
    plt.text(mean_index['index'], mean_index['e_dem_index'], country)

plt.title('Mean Proposed Democracy Index vs Mean Economist Index')
plt.xlabel('Mean Index')
plt.ylabel('Mean e_dem_index')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
C:\Users\pedro\AppData\Local\Temp\ipykernel_8780\16235499.py:1: FutureWarning:
The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a
future version, numeric_only will default to False. Either specify numeric_only
or select only columns which should be valid for the function.
  mean_values_by_country = sum_euro_c_data.groupby('cntry').mean()
```



As can be seen above, there seems to have been a noticeable improvement in the ranking and positioning of countries. Although the national means of Greece (GR) and Croatia (HR) in the proposed index are still better than that of the Economist's Democracy Index, this difference seems noticeably smaller than before, and these are the two countries who had a noticeable boost despite being less democratic. Italy (IT) also seems to be categorized slightly less democratic than what would be expected, but this difference is not as substantial as that of Greece and Croatia. Other important outliers we saw before, as North Macedonia (MK), Czechia (CZ), Belgium (BE) and Ireland (IE) are now much closer to their correct ranking than with the previous direct measure.

Clearly this proposed index to measure democratic values is far from perfect, it seems like a versatile tool in which many dimensions of democratic values can be captured and summarized into a single variable, making it useful for analysis in social sciences.

2. Which individual characteristics (sociodemographics) present a higher correlation with your support-for-democracy measure?

With regards to the sociodemographic variables highly correlated to the proposed democracy index above, all variables from the sociodemographics section will be introduced to see which have the strongest effects:

[9]:
```
sociodemographic_variables = ['index',
    'rshpsts', 'rshpsmk', 'rshpsgb', 'lvgptnea', 'dvrcdeva', 'marsts',␣
 ↪'marstmk', 'marstgb',
    'maritalb', 'chldhhe', 'domicil', 'edulvlb', 'eisced', 'edlvebe', 'edlvebg',
    'edlvdch', 'edlvehr', 'edlvdcz', 'edlvdee', 'edlvdfi', 'edlvdfr',␣
 ↪'edlvegr', 'edlvdahu',
    'edlvdis', 'edlvdie', 'edlveit', 'edlvdlt', 'edlvdme', 'edlvenl', 'edlvdmk',
```

```
    'edlvdpt', 'edlvesi', 'edlvdsk', 'educgb1', 'edubgb2', 'edagegb', 'eduyrs',␣
 ↪'pdwrk',
    'edctn', 'uempla', 'uempli', 'dsbld', 'rtrd', 'cmsrv', 'hswrk', 'dngoth',␣
 ↪'dngref',
    'dngdk', 'dngna', 'mainact', 'mnactic', 'crpdwk', 'pdjobev', 'pdjobyr',␣
 ↪'emplrel', 'emplno',
    'wrkctra', 'estsz', 'jbspv', 'njbspv', 'wkdcorga', 'iorgact', 'wkhct',␣
 ↪'wkhtot', 'nacer2',
    'tporgwk', 'isco08', 'wrkac6m', 'uemp3m', 'uemp12m', 'uemp5yr', 'mbtru',␣
 ↪'hincsrca',
    'hinctnta', 'hincfel', 'edulvlpb', 'eiscedp', 'edlvpebe', 'edlvpebg',␣
 ↪'edlvpdch',
    'edlvpehr', 'edlvpdcz', 'edlvpdee', 'edlvpdfi', 'edlvpdfr', 'edlvegr',␣
 ↪'edlvpdahu', 'edlvpdis',
    'edlvpdie', 'edlvpeit', 'edlvpdlt', 'edlvpdme', 'edlvenl', 'edlvpdmk',␣
 ↪'edlveno', 'edlvpdpt',
    'edlvpesi', 'edlvpdsk', 'edupcgb1', 'edupbgb2', 'edagepgb', 'pdwrkp',␣
 ↪'edctnp', 'uemplap',
    'uemplip', 'dsbldp', 'rtrdp', 'cmsrvp', 'hswrkp', 'dngothp', 'dngdkp',␣
 ↪'dngnapp', 'dngrefp',
    'dngnap', 'mnactp', 'crpdwkp', 'isco08p', 'emprelp', 'wkhtotp', 'edulvlfb',␣
 ↪'eiscedf',
    'edlvfebe', 'edlvfebg', 'edlvfdch', 'edlvfehr', 'edlvfdcz', 'edlvmdee',␣
 ↪'edlvmdfi', 'edlvmdfr',
    'edlvmegr', 'edlvfdahu', 'edlvfdis', 'edlvfdie', 'edlvfeit', 'edlvfdlt',␣
 ↪'edlvfdme', 'edlvfenl',
    'edlvfdmk', 'edlvfeno', 'edlvfdpt', 'edlvmesi', 'edlvfdsk', 'edufcgb1',␣
 ↪'edufbgb2', 'edagemgb',
    'emprf14', 'occm14b', 'atncrse', 'anctry1', 'anctry2', 'regunit', 'region'
]

sociodemographic_data = sum_euro_c_data[sociodemographic_variables]

correlation_matrix = sociodemographic_data.corr(numeric_only=True)

correlation_with_democracy = correlation_matrix['index']

sorted_correlation = correlation_with_democracy.abs().
 ↪sort_values(ascending=False)

print("Top correlated sociodemographic variables with personal democracy index:
 ↪")
print(sorted_correlation.head(16))
```

```
Top correlated sociodemographic variables with personal democracy index:
index       1.000000
```

```
edlveno     0.337729
edlvdpt     0.284096
edlvehr     0.237181
edlvdch     0.225660
edlvdee     0.196029
hincfel     0.189925
edlvebg     0.174549
atncrse     0.172138
hinctnta    0.166196
edlvdis     0.162695
mbtru       0.145104
pdwrk       0.132434
edlvfebe    0.131819
edlvdcz     0.130006
pdjobev     0.129377
Name: index, dtype: float64
```

When checking these variables' ranking with regards to their correlation to the index, the main sociodemographic consideration seems to be education ('edlvxx' variables) and preparation ('atncrse') of indivdiduals in their different countries. This result is consistent with the literature regarding democratic values, as various studies (Bratton and Mattes 2001; Norris and Inglehart, 2004) find that individuals who tend to get more education, find out about the advantages of this form of government over the alternatives and, therefore, prefer democracy.

Other variables that seem to be correlated with support for demcoracy is satisfaction with income ('hincfel') and income level ('hinctnta'). This seems to show two trends: being at different levels of income tends to affect a person's disposition towards ddemocracy, while their perceived (personal) satisfaction with their economic situation also seems to have an effect on their disposition towards democracy.

Finally, individuals who are engaging in economic activities, as being currently engaged in paid work ('pdwrk') being part of a trade union ('mbtru') or simply having had a job in the past ('pdjobev') also seems to be among the most correlated with support for democracy. All these findings regarding wealth (earning, satisfaction and employment status) seem to reflect important democratic principles (Manin, 1997): a tenant in the foundation of modern representative democracy was people's committment with citizenship, which was reflected in their earning and income. In other words, since individuals who had money and property seemed were the ones most affected by democratic decisions (via taxes and economic loss), it was mandated for the earliest voters to have property and surpass an income threshold for them to engage in democracy. Although this belief could seem antiquated, perhaps it reflects a truth applicable to our modern context: individuals who have something to lose or that have experienced the costs of working could be the largest proponents of democracy.

Finally, it is reasonable to see if there are additional sociodemographic variables that could be correlated with the proposed democracy index.

Performing a multivariate regression estimation, we get the following:

[10]:
```
X = sum_euro_c_data[['trstep', 'hinctnta', 'dscrgrp','rlgdgr','hmsacld']]
y = sum_euro_c_data['index']
```

```
X = sm.add_constant(X)

model = sm.OLS(y, X).fit()

print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  index   R-squared:                       0.098
Model:                            OLS   Adj. R-squared:                  0.098
Method:                 Least Squares   F-statistic:                     303.1
Date:                Sun, 07 Apr 2024   Prob (F-statistic):          6.07e-309
Time:                        12:37:34   Log-Likelihood:                 6414.7
No. Observations:               13889   AIC:                         -1.282e+04
Df Residuals:                   13883   BIC:                         -1.277e+04
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.6737      0.005    128.481      0.000       0.663       0.684
trstep         0.0029      0.001      5.498      0.000       0.002       0.004
hinctnta       0.0075      0.001     14.961      0.000       0.007       0.008
dscrgrp        0.0245      0.005      5.012      0.000       0.015       0.034
rlgdgr        -0.0049      0.000    -11.465      0.000      -0.006      -0.004
hmsacld       -0.0235      0.001    -24.898      0.000      -0.025      -0.022
==============================================================================
Omnibus:                      323.127   Durbin-Watson:                   1.859
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              301.262
Skew:                          -0.317   Prob(JB):                     3.82e-66
Kurtosis:                       2.655   Cond. No.                         40.5
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

These results show some patterns that could seem expected. All else equal: individuals who are more democratic tend to have a positive and statistically significant correlation to trusting large international institutions, in this case the European Parliament (trstep); individuals with more wealth tend to be more democratic (hinctnta), as mentioned in the discussion above. Additionally, individuals who come from a discriminated group (hmsacld') also seem to be less democratic. On another note, individuals who favor progressive agendas ('dscrgrp'), as rights for the LGBTQ+ group, seem to also seem to score high in democratic preferences, which is consistent with research on these trends (Pew Research Center, 2021).

Finally, we also observe that individuals who are more religious ('rlgdgr') tend to be less in favor of democracy, something that could be reflecting a sociological and institutional argument (Schmitt,

[1922] 1985): since individuals who are religious are used to respect and obey a set of principles and rules imposed from a deity, they are less prone to accept a system of government like democracy in which the laws and principles are determined by mere popular preferences, rather than a more robust and credible authority.

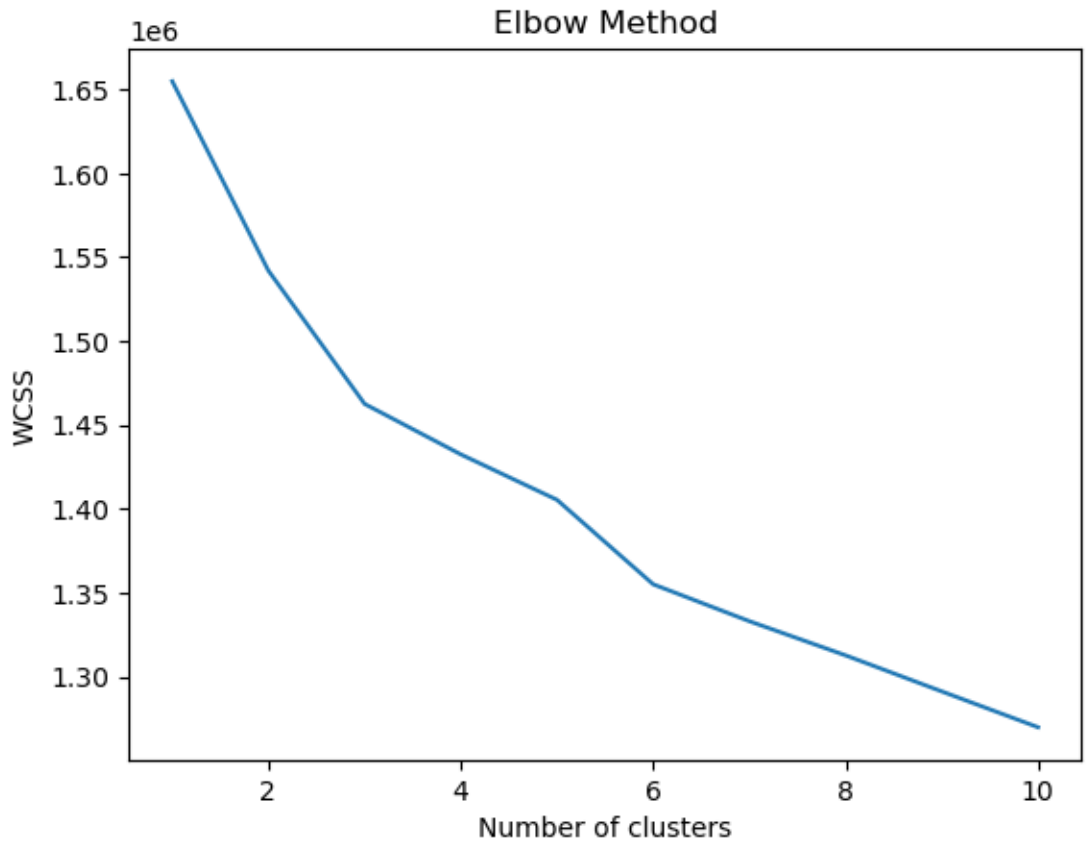
3. We would like to group individuals according to their political views. Could you perform a Cluster Analysis and help us group individuals?

[11]:
```
selected_columns = [
    "polintr", "psppsgva", "actrolga", "psppipla", "cptppola",
    "trstprl", "trstlgl", "trstplc", "trstplt", "trstprt",
    "trstep", "trstun", "trstsci", "vote", "contplt", "donprty", "badge",
    "sgnptit", "pbldmna", "bctprd", "pstplonl", "volunfp",
    "clsprty", "prtdgcl", "lrscale", "stflife", "stfeco",
    "stfgov", "stfdem", "stfedu", "stfhlth", "gincdif",
    "freehms", "hmsfmlsh", "hmsacld", "euftf", "lrnobed",
    "loylead", "imsmetn", "imdfetn", "impcntr", "imbgeco",
    "imueclt", "imwbcnt"
]

selected_data = euro_c_data[selected_columns]

selected_data.fillna(0, inplace=True)

scaler = StandardScaler()
scaled_data = scaler.fit_transform(selected_data)

kmeans = KMeans(n_clusters=3)
kmeans.fit(scaled_data)

euro_c_data['cluster_label'] = kmeans.labels_

cluster_centers = pd.DataFrame(scaler.inverse_transform(kmeans.
 ↪cluster_centers_), columns=selected_data.columns)
print("Cluster Centers:")
print(cluster_centers)

print("Distribution of data points in each cluster:")
print(euro_c_data['cluster_label'].value_counts())
```

```
C:\Users\pedro\AppData\Local\Temp\ipykernel_8780\3730958773.py:15:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  selected_data.fillna(0, inplace=True)

Cluster Centers:
```

```
    polintr  psppsgva  actrolga  psppipla  cptppola    trstprl    trstlgl  \
0  2.271784  2.740730  2.688138  2.710974  2.705784   5.954616   6.511505
1  3.492205  3.325909  2.341500  2.936897  2.409800  33.365999  30.086860
2  3.134070  1.868837  1.649051  1.712519  1.700682   4.191143   4.978439

     trstplc    trstplt    trstprt  …   hmsacld      euftf   lrnobed  \
0   7.043250   4.713800   4.831152  …  2.305865   8.279107  2.674067
1  18.242762  30.093541  34.834447  …  4.222717  51.098738  2.869339
2   5.978122   3.085716   3.098029  …  3.817788  10.536173  2.258680

    loylead   imsmetn   imdfetn   impcntr    imbgeco    imueclt    imwbcnt
0  3.259443  1.753013  1.987890  2.065740   7.253215   7.444553   7.234531
1  4.572383  3.311804  3.595397  3.622866  34.789161  31.803267  32.576095
2  3.094065  2.607039  3.043545  3.103366   5.859959   6.011256   6.170322

[3 rows x 44 columns]
Distribution of data points in each cluster:
2    18927
0    17337
1     1347
Name: cluster_label, dtype: int64
```

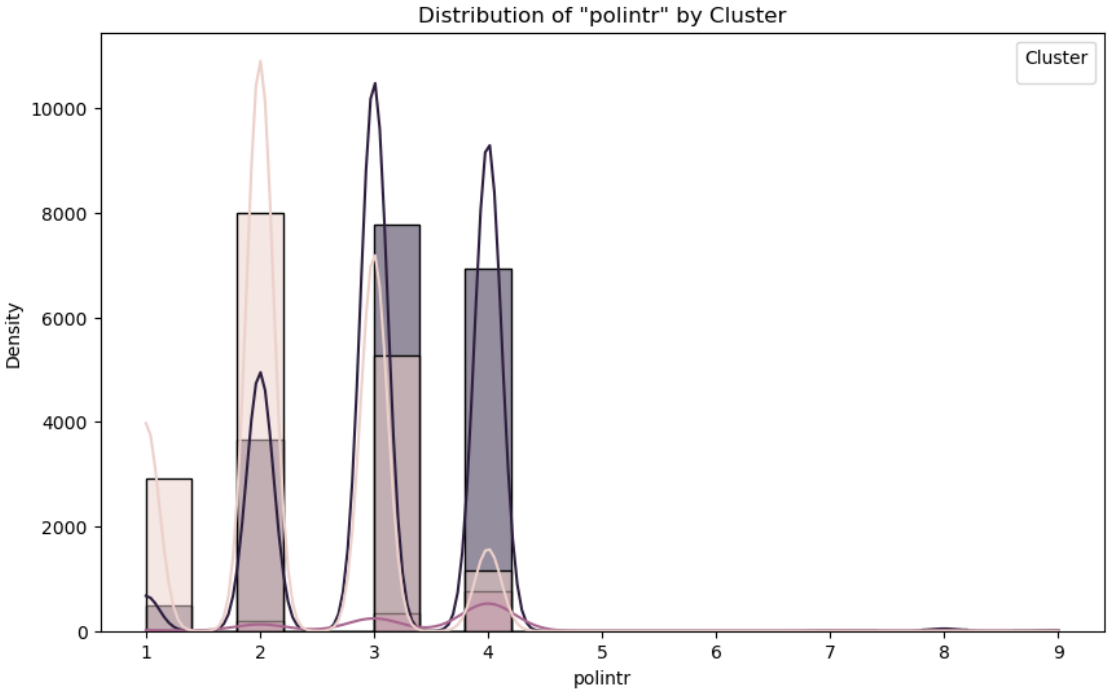
[12]:
```
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(scaled_data)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



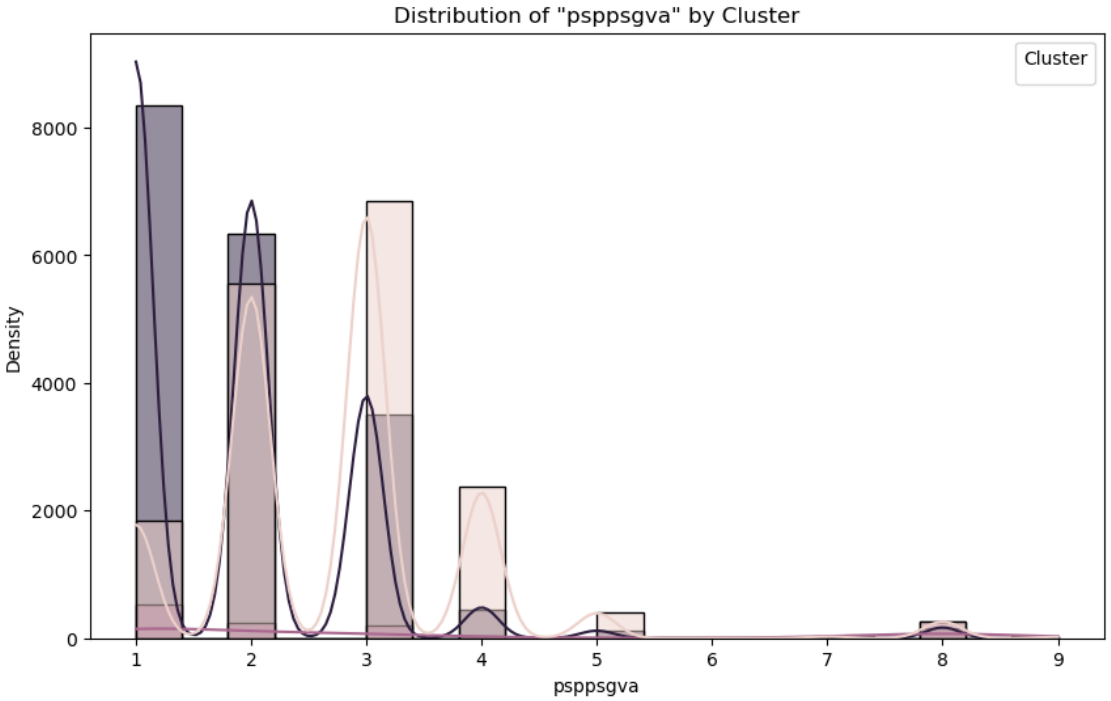
Based on the diagram above (elbow method), it would seem that using 3 clusters wwas feasible for this data, as it seems that adding more than 3 clusters leads to diminishing returns in terms of reducing the sum of the squared distances (7 more clusters offer a smaller reduction in distance -.15- compared to only 3 clusters).

[13]:
```
variables_of_interest = ['polintr', 'psppsgva', 'actrolga', 'psppipla',␣
 ↪'cptppola']

for variable in variables_of_interest:
    plt.figure(figsize=(10, 6))
    sns.histplot(data=euro_c_data, x=variable, hue='cluster_label', kde=True,␣
 ↪bins=20)
    plt.title(f'Distribution of "{variable}" by Cluster')
    plt.xlabel(variable)
    plt.ylabel('Density')
    plt.legend(title='Cluster')
    plt.show()
```

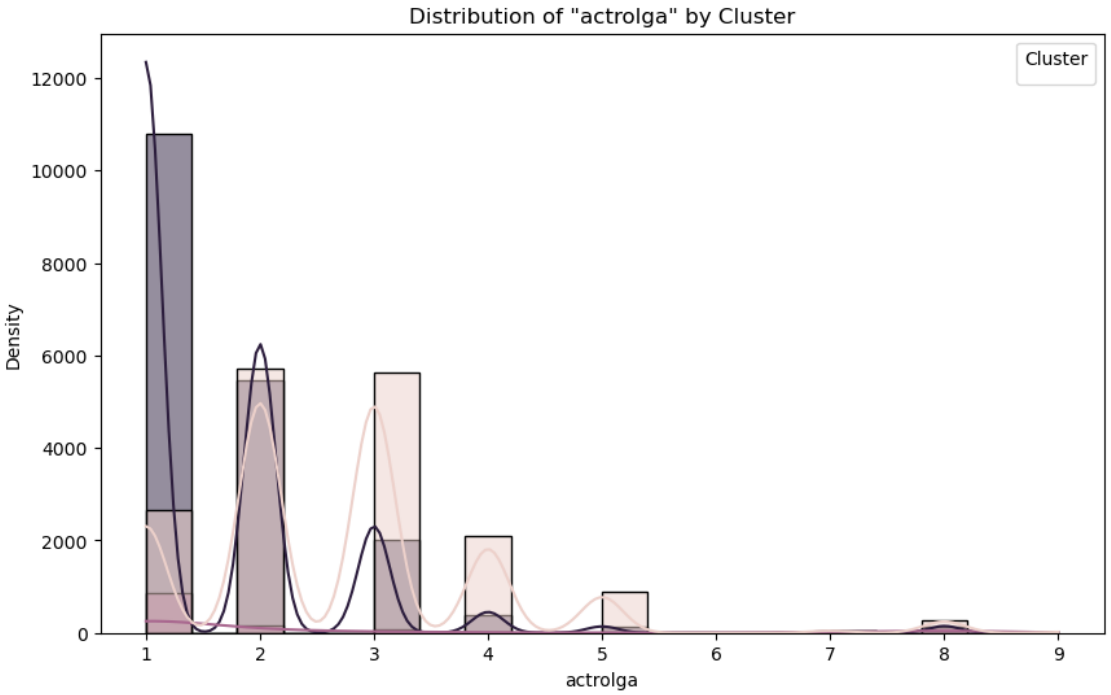
```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



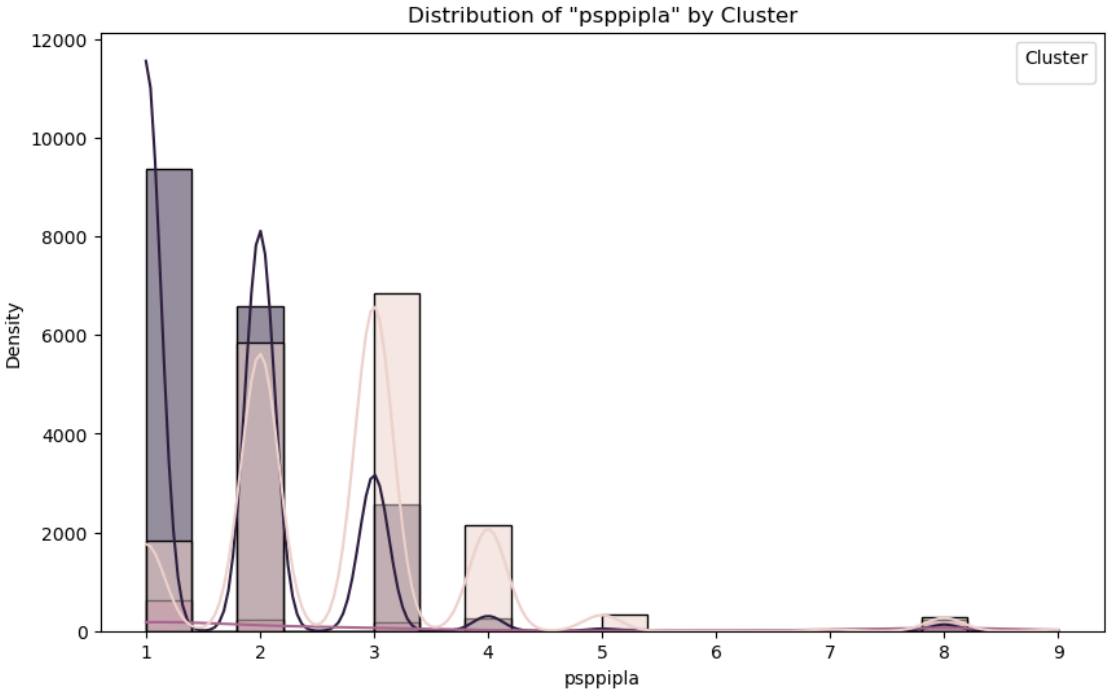

```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



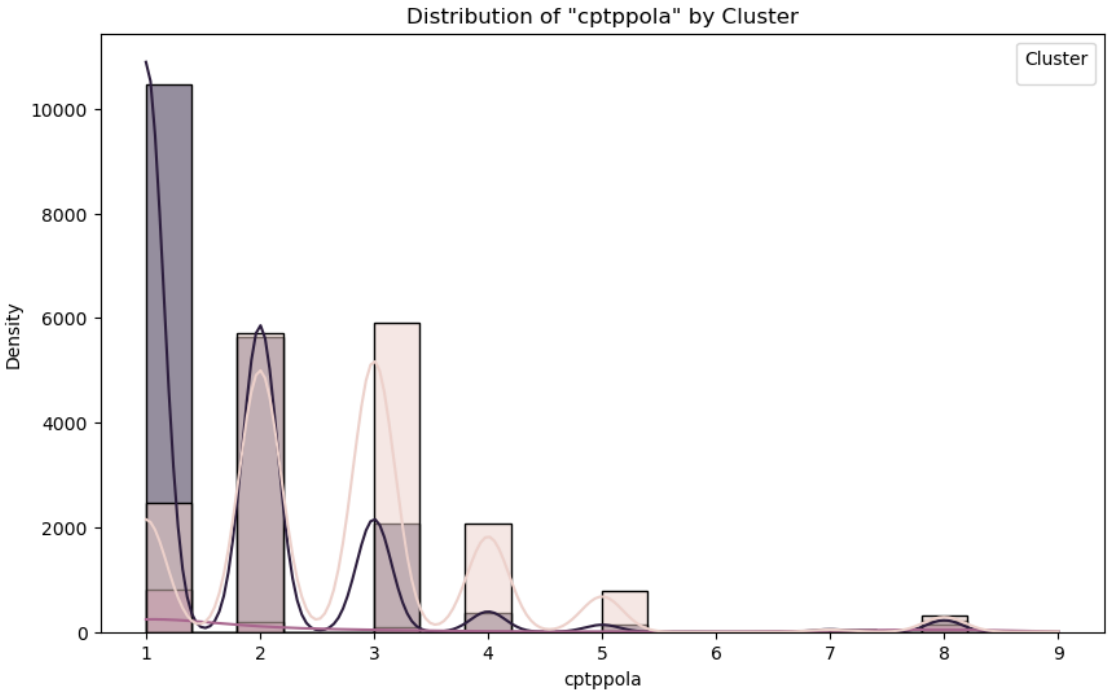
```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```




```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```



While seeing the distribution of variables based on clusters, it would seem that the clustering technique offers mixed results. On one hand, there does seem to be a reasonable difference between the means per cluster, the observations of clearer color clusters seem to be more apart (at the opposite distribution from left to right) from darker color clusters among the different visualizations above, which shows differentiation among these. On the other hand, most clusters do have a sizeable amount of their elements between two or more of the distributions in the visualizations below, showing that perhaps the separation of 3 clusters among these variables didn't differentiate these indivduals in a very rigorous manner. Overall, it seems workable, but perfectable.