Project Report

By: S.M SURYAPRAKSH

Abstract–

The purpose of this report is to document the first steps losses the data science process, including data cleaning and exploration, on a given data file(s). The data file(s) were loaded into memory, processed, and analysed using appropriate steps in Jupyter notebook, which is part of the Anaconda version specified in the canvas announcement. This assignment is intended to provide practical experience in the typical initial steps of the data science process. The report will detail the methods and techniques used for data cleaning, processing, and analysis, along with the results obtained from the analysis.

INTRODUCTION -

Data science has become an essential tool in making informed decisions in today's data-driven world. One of the key first steps in the data science process is data cleaning and exploration. In this assignment, we will explore a data file and carry out the initial steps of the data science process. The objective of this assignment is to load a data file into memory, clean, process and analyse it. We will be using Jupyter Notebook, a powerful tool for data science, to accomplish this task. The data file(s) we will be working with might contain inconsistencies, errors, or missing values. Therefore, we will need to clean and process the data before analysing it. Cleaning the data involves removing or correcting any errors or inconsistencies. Once the data is cleaned, we can then explore it to identify patterns, trends, and relationships. This exploration will help us to identify any potential insights or trends that may be hidden within the data. In conclusion, this assignment is designed to give practical experience with the initial steps of the data science process. By exploring and cleaning the data, we can uncover valuable insights and make informed decisions.

METHODOLOGY

Data science involves a systematic process of collecting, cleaning, analysing, and interpreting data to extract insights and make informed decisions. In this project, we will follow a similar approach to explore and analyse three datasets. Our methodology will involve five key steps: loading the datasets, cleaning and exploring each dataset separately, merging the datasets into a single data frame, analysing the merged data, and finally summarizing our findings in a report.

1. Loading the Datasets:

Our first step is to load the three datasets into separate Pandas Data Frames. Pandas is a powerful data manipulation library in Python that provides functions for data cleaning and exploration. We will use the read_csv() function from pandas to read the CSV files.

2. Data Cleaning and Exploration:

Once we have loaded the datasets, our next step is to clean and explore each dataset separately. We will start by checking for missing values and removing any duplicates. If any inconsistencies or errors are found, we will need to decide how to handle them. In some cases, we may need to impute missing values, correct errors, or remove irrelevant data. After cleaning the data, we will then carry out exploratory data analysis (EDA) to better understand the data and identify any patterns, trends, or relationships. During the EDA, we will create visualizations using Matplotlib and Seaborn libraries to help us better understand the data. We will look at various statistical measures such as mean, median, mode, standard deviation, variance, etc. to get a better understanding of the data.

3. Merging Datasets:

After cleaning and exploring each dataset, we will merge them into a single Data Frame using the merge () function from Pandas. We will identify the common columns between the datasets and use them to merge the datasets. We will use the merge function to combine the datasets based on common fields in the datasets. This will allow us to analyse the data as a whole and identify any potential insights or trends.

4. Data Analysis:

With our merged dataset, we will perform further analysis to extract valuable insights from the data. We will use various statistical techniques such as correlation analysis, regression analysis, etc., to identify relationships between variables and understand the impact of different factors on the outcomes. For example, we may identify correlations between the sales of a product and the marketing budget spent on that product. We may also identify trends in the data such as seasonal variations in sales or changes in customer behaviour over time. By analysing the data in detail, we will be able to identify key drivers of business success and help guide decision-making.

5. Conclusion:

Finally, we will summarize our findings and conclusions in the report. We will highlight any important insights, trends, or patterns that we discovered during our analysis. We will also discuss any limitations or caveats to our analysis and provide recommendations for future research.

For example, we may recommend that a company increase its marketing budget during certain times of the year to take advantage of seasonal variations in customer behaviour. We may also recommend that the company analyse the impact of different products on sales to identify the most profitable products to focus on.

In conclusion, by following this methodology, we will be able to load, clean, explore, and analyse three separate datasets and merge them into a single Data Frame for further analysis. Through our analysis, we will be able to extract valuable insights and better understand the relationship between different variables. By presenting our findings in a clear and concise report, we will be able to provide valuable insights to guide decision-making and help drive business success.

DATAPROCESSING

Data Pre-processing:

After Database creation, Data pre-processing helps to remove noise, missing values, and inconsistencies. Missing values are replaced with NULL. Also, each attribute data is discretized in order to make it appropriate for further analysis.

Data Cleaning:

One critical aspect of data cleaning is treating missing values, which could affect the accuracy of our analysis. We identified missing values in the dataset and used various techniques to impute missing values. These techniques included mean, median, and mode imputation, and we selected the most appropriate method based on the distribution and characteristics of the data.

In conclusion, the use of EDA, data wrangling techniques, and data cleaning methods are essential steps in any data analysis project. The methods discussed in this project help to identify patterns, relationships, and outliers in the data and ensure that the data is cleaned and transformed correctly for subsequent analysis. The analysis of the data provides valuable insights into the characteristics of the data, which can be used to inform decision-making processes. Overall, this project demonstrates the importance of understanding and preparing the data for analysis to derive meaningful insights and conclusions.

Code :

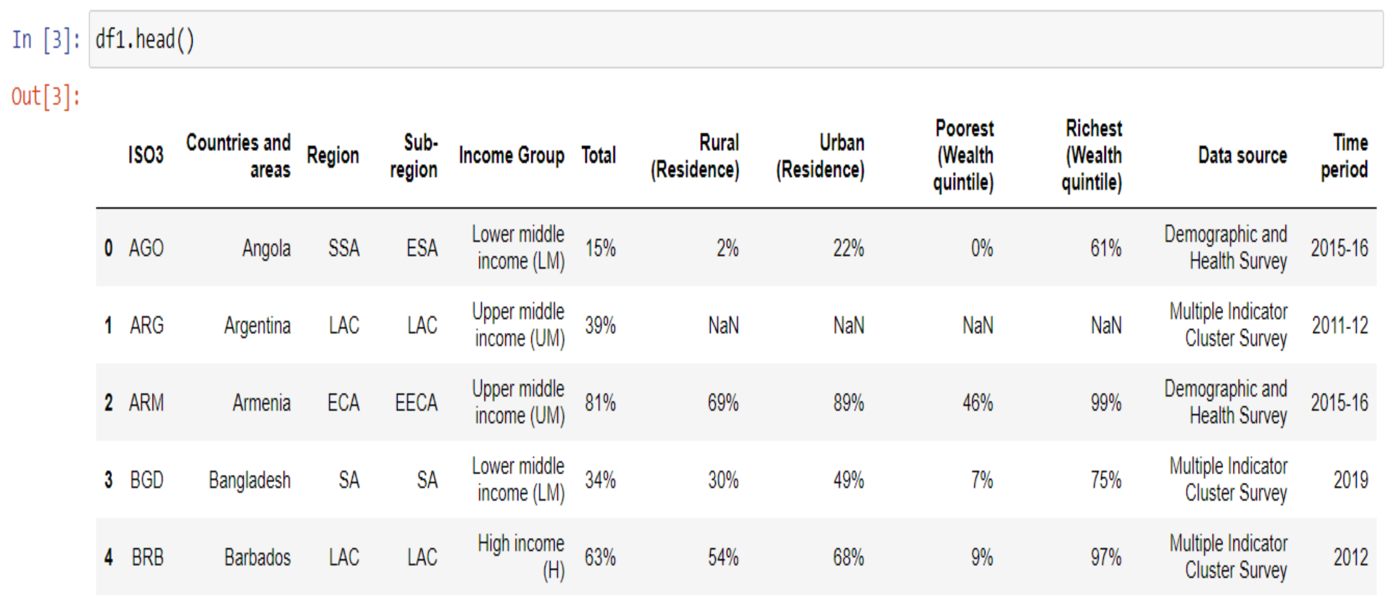
1. Importing the Required Libraries

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

2. Reading the Datasets

```
 df1 = pd.read_csv('Primary.csv',encoding='ISO-8859-1',header=1)
df2 = pd.read_csv('Secondary.csv',encoding='ISO-8859-1',header=1)
df3 = pd.read_csv('Total School Age.csv',encoding='ISO-8859-1',header=1)
```

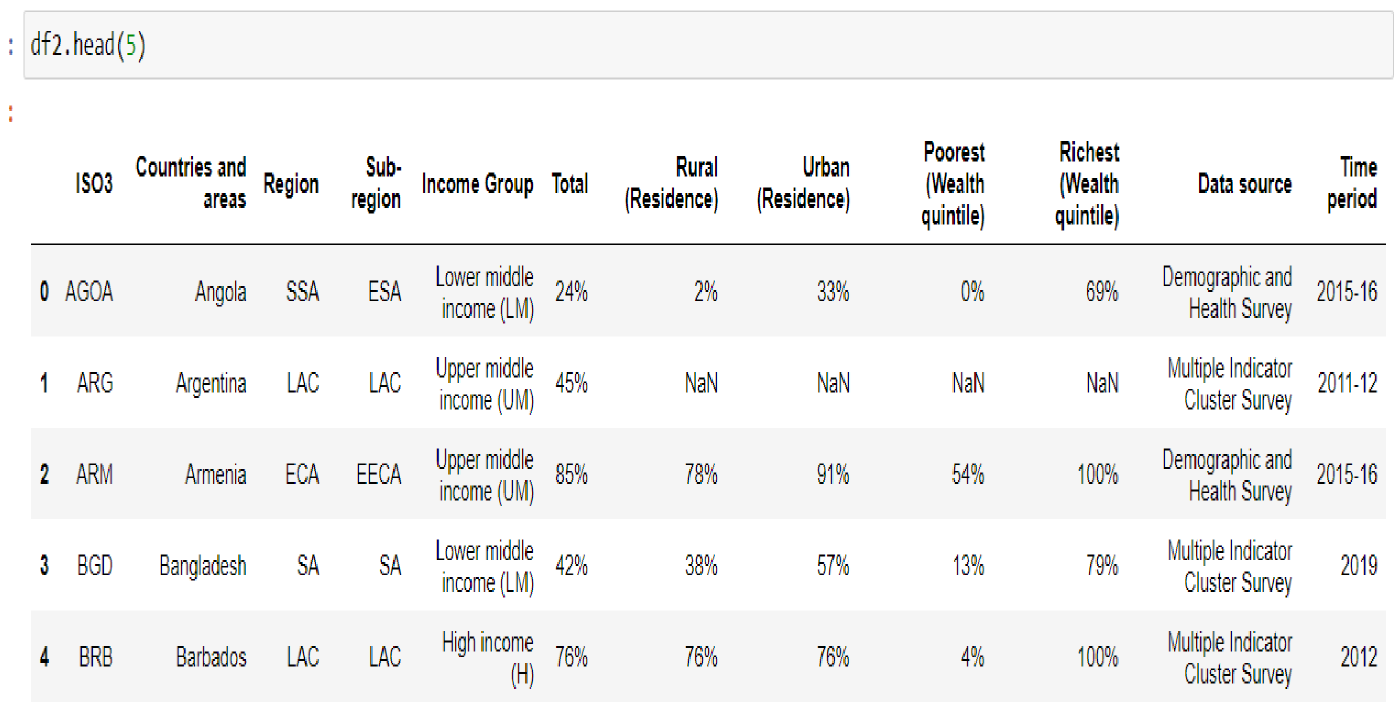
3. Looking into Top 5 Records



```
In [3]: df1.head()
```

Out[3]:

	ISO3	Countries and areas	Region	Sub-region	Income Group	Total	Rural (Residence)	Urban (Residence)	Poorest (Wealth quintile)	Richest (Wealth quintile)	Data source	Time period
0	AGO	Angola	SSA	ESA	Lower middle income (LM)	15%	2%	22%	0%	61%	Demographic and Health Survey	2015-16
1	ARG	Argentina	LAC	LAC	Upper middle income (UM)	39%	NaN	NaN	NaN	NaN	Multiple Indicator Cluster Survey	2011-12
2	ARM	Armenia	ECA	EECA	Upper middle income (UM)	81%	69%	89%	46%	99%	Demographic and Health Survey	2015-16
3	BGD	Bangladesh	SA	SA	Lower middle income (LM)	34%	30%	49%	7%	75%	Multiple Indicator Cluster Survey	2019
4	BRB	Barbados	LAC	LAC	High income (H)	63%	54%	68%	9%	97%	Multiple Indicator Cluster Survey	2012



```
: df2.head(5)
```

:

	ISO3	Countries and areas	Region	Sub-region	Income Group	Total	Rural (Residence)	Urban (Residence)	Poorest (Wealth quintile)	Richest (Wealth quintile)	Data source	Time period
0	AGOA	Angola	SSA	ESA	Lower middle income (LM)	24%	2%	33%	0%	69%	Demographic and Health Survey	2015-16
1	ARG	Argentina	LAC	LAC	Upper middle income (UM)	45%	NaN	NaN	NaN	NaN	Multiple Indicator Cluster Survey	2011-12
2	ARM	Armenia	ECA	EECA	Upper middle income (UM)	85%	78%	91%	54%	100%	Demographic and Health Survey	2015-16
3	BGD	Bangladesh	SA	SA	Lower middle income (LM)	42%	38%	57%	13%	79%	Multiple Indicator Cluster Survey	2019
4	BRB	Barbados	LAC	LAC	High income (H)	76%	76%	76%	4%	100%	Multiple Indicator Cluster Survey	2012

In [5]:
```
df3.head(5)
```

Out[5]:

	ISO3	Countries and areas	Region	Sub-region	Income Group	Total	Rural (Residence)	Urban (Residence)	Poorest (Wealth quintile)	Richest (Wealth quintile)	Data source	Time period
0	DZA	Algeria	MENA	MENA	Upper middle income (UM)	24%	9%	32%	1%	77%	Multiple Indicator Cluster Survey	2018-19
1	AGO	Angola	SSA	ESA	Lower middle income (LM)	17%	2%	24%	0%	62%	Demographic and Health Survey	2015-16
2	ARG	Argentina	LAC	LAC	Upper middle income (UM)	40%	NaN	NaN	NaN	NaN	Multiple Indicator Cluster Survey	2011-12
3	ARM	Armenia	ECA	EECA	Upper middle income (UM)	81%	71%	88%	47%	99%	Demographic and Health Survey	2015-16
4	BGD	Bangladesh	SA	SA	Lower middle income (LM)	37%	33%	52%	9%	76%	Multiple Indicator Cluster Survey	2019

4. Looking Data Types

In [6]:
```
print('Number of Rows:',df1.shape[0])
print('Number of Columns:',df1.shape[1])
```

```
Number of Rows: 87
Number of Columns: 12
```

observation:

1. Therefore we can see that there are about 87 records and 12 columns present.
2. We can see that there 12 columns all are kind of Categorical while looking the top 5 records will make sure in further analysis.
3. Encoding is done for all these datasets and header is given for the better understanding of the datas we have.
4. So the Next step is to clean the DataFrame 1 and Treat all the Null values.
5. Therefore we can do the visualization for the required stuffs and we can draw some insights.

In [7]:
```
print('Categorical Columns:',df1.select_dtypes(exclude=np.number).columns)
print('----------------------------------------------------------------------')
print('Numerical Columns:',df1.select_dtypes(include=np.number).columns)
```

```
Categorical Columns: Index(['ISO3', 'Countries and areas', 'Region', 'Sub-region', 'Income Group',
       'Total', 'Rural (Residence)', 'Urban (Residence)',
       'Poorest (Wealth quintile)', 'Richest (Wealth quintile)', 'Data source',
       'Time period'],
      dtype='object')
----------------------------------------------------------------------
Numerical Columns: Index([], dtype='object')
```

Observation:

1. From the above dataset 1 we can see that there is no presence of Numerical Columns.
2. All the 12 columns are in the Categorical types but we can see that Total', 'Rural (Residence)', 'Urban (Residence)','Poorest (Wealth quintile)', 'Richest (Wealth quintile) are needed to be converted into the Numerical type and therefore we do the required stuffs to convert these into Numerical Stuffs.

5. Conversion of Data Types to Required Data Types

```
In [8]: df1['Total'] = df1['Total'].str.split('%',expand=True)[0].astype(float)
```

```
In [9]: df1['Rural (Residence)'] = df1['Rural (Residence)'].str.split('%',expand=True)[0].astype(float)
```

```
In [10]: df1['Urban (Residence)'] = df1['Urban (Residence)'].str.split('%',expand=True)[0].astype(float)
```

```
In [11]: df1['Poorest (Wealth quintile)'] = df1['Poorest (Wealth quintile)'].str.split('%',expand=True)[0].astype(float)
```

```
In [12]: df1['Richest (Wealth quintile)'] = df1['Richest (Wealth quintile)'].str.split('%',expand=True)[0].astype(float)
```

```
In [13]: print('Categorical Columns:',df1.select_dtypes(exclude=np.number).columns)
         print('----------------------------------------------------------------------')
         print('Numerical Columns:',df1.select_dtypes(include=np.number).columns)

         Categorical Columns: Index(['ISO3', 'Countries and areas', 'Region', 'Sub-region', 'Income Group',
                'Data source', 'Time period'],
               dtype='object')
         ----------------------------------------------------------------------
         Numerical Columns: Index(['Total', 'Rural (Residence)', 'Urban (Residence)',
                'Poorest (Wealth quintile)', 'Richest (Wealth quintile)'],
               dtype='object')
```

Observation:

1. Therefore we have converted the Total', 'Rural (Residence)', 'Urban (Residence)','Poorest (Wealth quintile)', 'Richest (Wealth quintile) to the Numerical type with the help of split method and I have removed the '%' and then converted these to float type.
2. We can see that the type of this converted variables in the Numerical Column and therefore the variables are converted to required type and therefore we can proceed further.

6. Checking Null Values

```
In [25]: df1.isnull().sum()

Out[25]: Region                        0
         Sub-region                    0
         Income Group                  0
         Total                         0
         Rural (Residence)            11
         Urban (Residence)             8
         Poorest (Wealth quintile)    18
         Richest (Wealth quintile)    21
         Data source                   0
         Time period                   4
         dtype: int64
```

observation:

1. The Rural(Residence) as 11,Urban(Residence)as 8,Poorest(Wealth quintile)as 18,Richest(Wealth quintile)as 21,Time period as 4 Null values and these are needed to be imputed.
2. Before we impute we need to look into the skew for imputing the Mean or Median values for the Numerical Columns.

7. Conversion of Null Values

Observation:

1. As we can see that they all are skewed for Numerical values and we can't impute the Missing values with the Mean value.
2. The black Line represents the Median value and the Red line represents the Mean values.
3. The Line showcases us that how the variables are present in the dataset we have and we can impute the nulls via but if we impute the Nulls like this the values that are filled either be a mean or median value but this doesn't make sense.
4. So we will use groupby and then impute the Necessary values to the columns.
5. By this method we can have a better value than imputing the null values from the columns's Median or Mean.

```
In [27]: df1['Urban (Residence)'] = df1.groupby('Region')['Urban (Residence)'].fillna(df1['Urban (Residence)'].median()).values
```

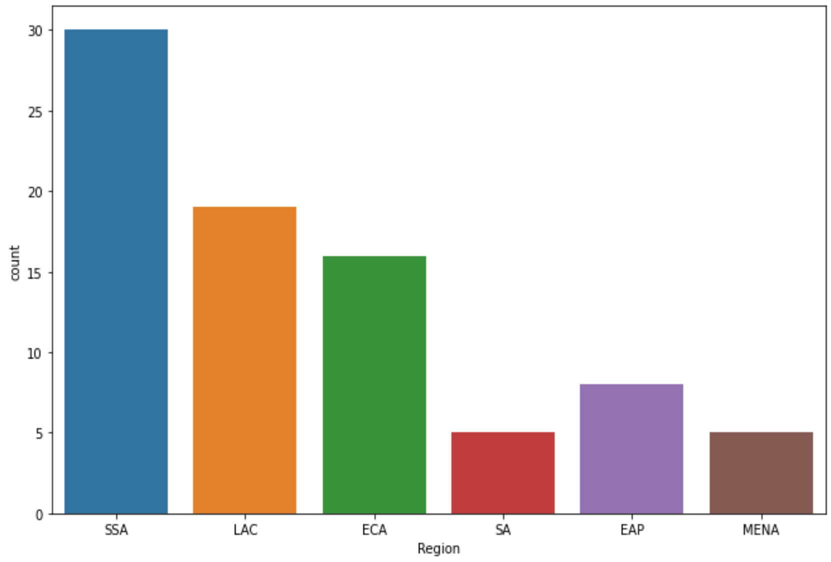
```
In [28]: df1['Poorest (Wealth quintile)'] = df1.groupby('Region')['Poorest (Wealth quintile)'].fillna(df1['Poorest (Wealth quintile)'].mec
```

```
In [29]: df1['Richest (Wealth quintile)'] = df1.groupby('Region')['Richest (Wealth quintile)'].fillna(df1['Richest (Wealth quintile)'].mec
```

```
In [30]: df1['Rural (Residence)'] = df1.groupby('Region')['Rural (Residence)'].fillna(df1['Rural (Residence)'].median()).values
```

EDA ANALYSIS:

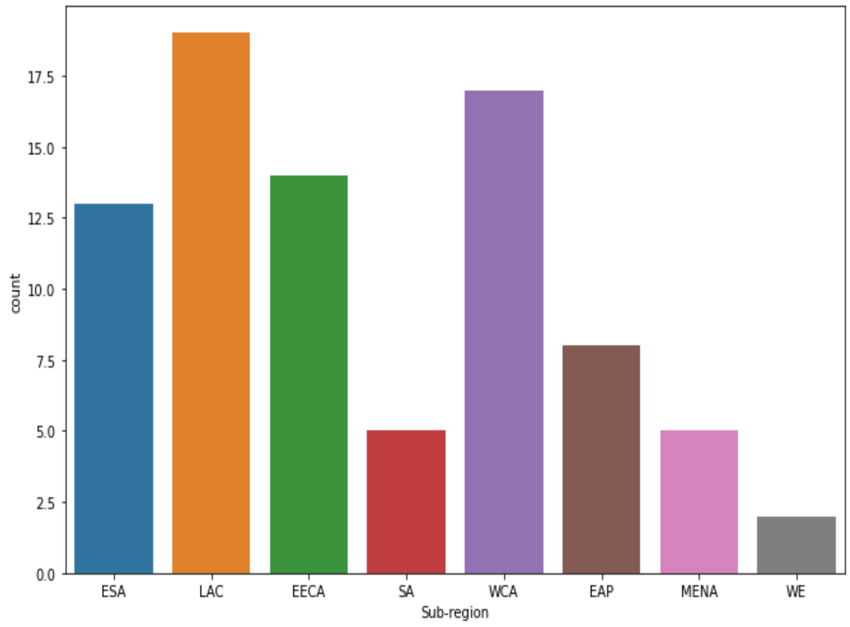
In [35]:
```python
plt.figure(figsize=(11,7))
sns.countplot(df1['Region'])
plt.show()
```



Inference:

1. The SSA is the largest count and therefore we have the LAC as the second Region in this dataFrame.
2. This shows that the first Region in this df1 is the SSA which means most the members around 30 are from this and then we have LAC in the second where the count is around 20.
3. The Last Region in this dataFrame is SA with the count of 5 which indicate that least members are from SA.

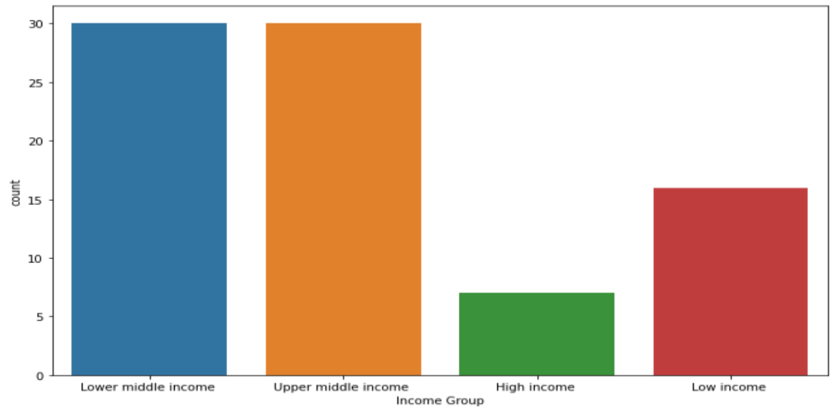
In [36]:
```python
plt.figure(figsize=(11,7))
sns.countplot(df1['Sub-region'])
plt.show()
```



Inference:

1. The LAC count is more in the sub-Region category and then comes the WCA where the count is quite Lesser in terms of the LAC.
2. Then we have in the last is the WE where the count is less than 2.5 and thus we can see it via the countplot.
3. The most important Sub-Region are LAC,WCA and EECA.

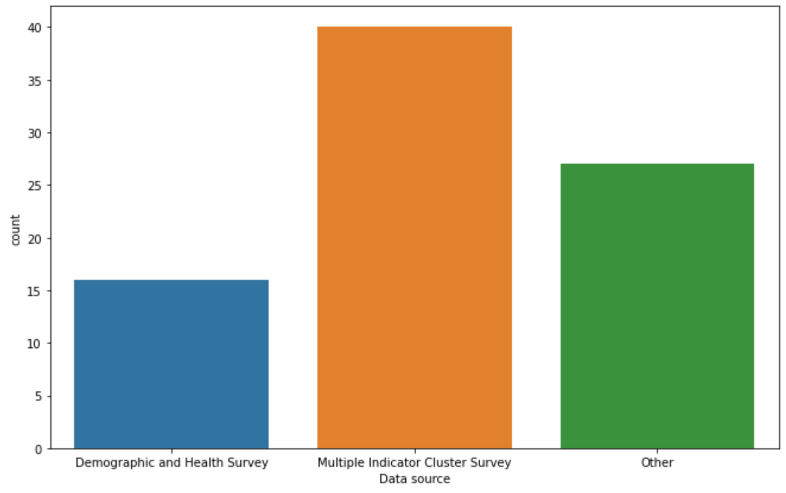
In [37]:
```python
plt.figure(figsize=(11,7))
sns.countplot(df1['Income Group'])
plt.show()
```



Inference:

1. It is very suprising to see that Lower Middle Income and the Upper Middle Income are falling into the same count around 30Each.
2. Then we have Low Income in the 3rd place where the count is lesser than 20.
3. Then we have High Income category where the count is too low than other 3 category and the count is also lower than 10.
4. Therefore we can conclude that we have a DataFrame where most people are not in High Income Category.

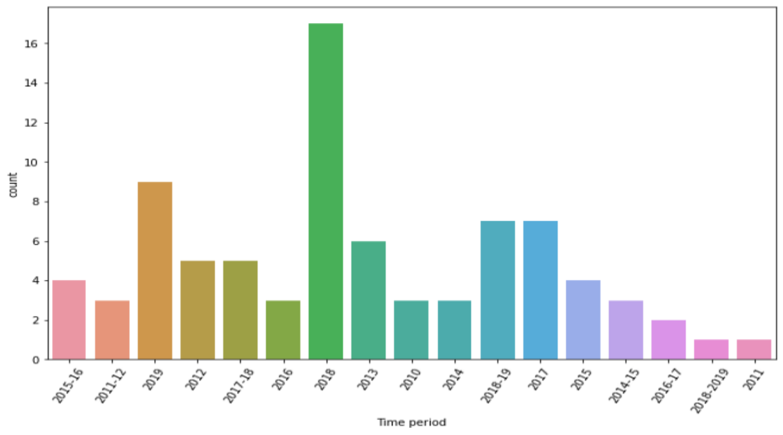
In [38]:
```python
plt.figure(figsize=(11,7))
sns.countplot(df1['Data source'])
plt.show()
```



Inference:

1. The DataSource is where the Data is collected and most of the datas are collected via Multiple Indicator Cluster Survey.
2. Then we have the Demographic and Health Survey; In this 2 Data Source almost all the Datas are collected.

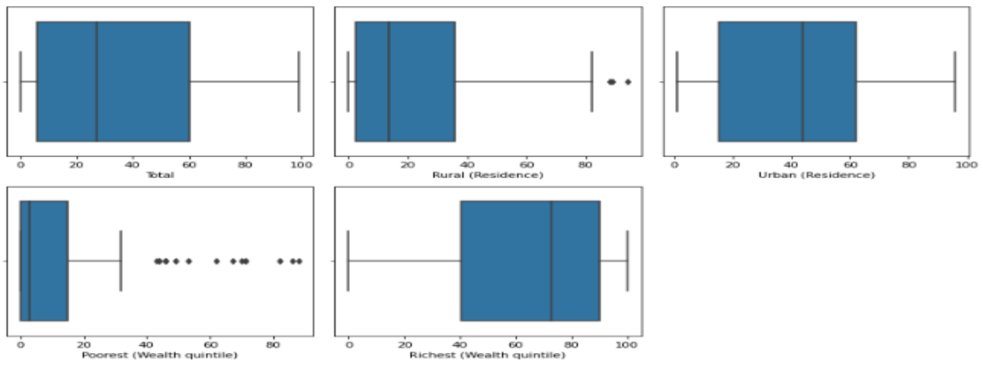
In [39]:
```python
plt.figure(figsize=(11,7))
sns.countplot(df1['Time period'])
plt.xticks(rotation=60)
plt.show()
```



Inference:

1. There are many Time Period but the 2018 Time Period has the most number of the count and it is around 16+ count and then next time period is that we have 2019 and it contributes around 8+ count.
2. The least is the 2011 count and it has less than 2 count and same for the 2018-2019 Time Period also.

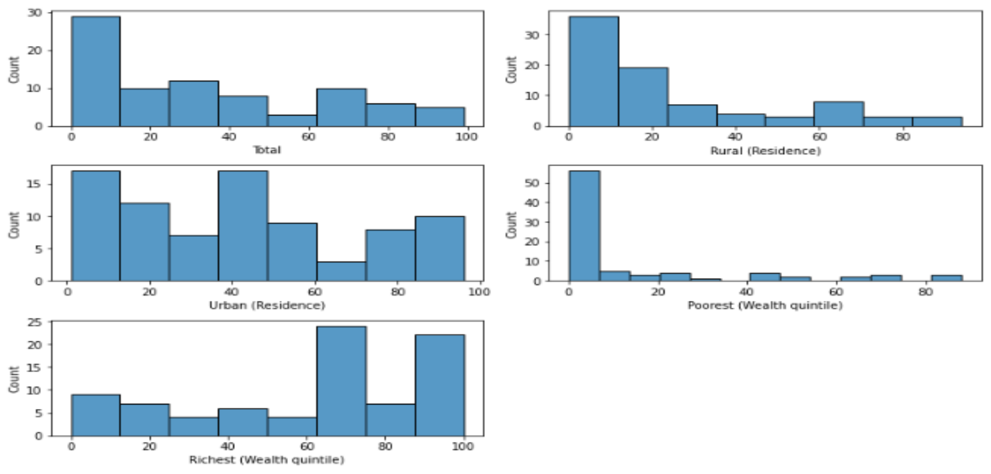
```
In [40]: it=1
         plt.figure(figsize=(11,7))
         for i in df1.select_dtypes(include=np.number):
             plt.subplot(2,3,it)
             sns.boxplot(df1[i])
             it=it+1
         plt.tight_layout()
         plt.show()
```



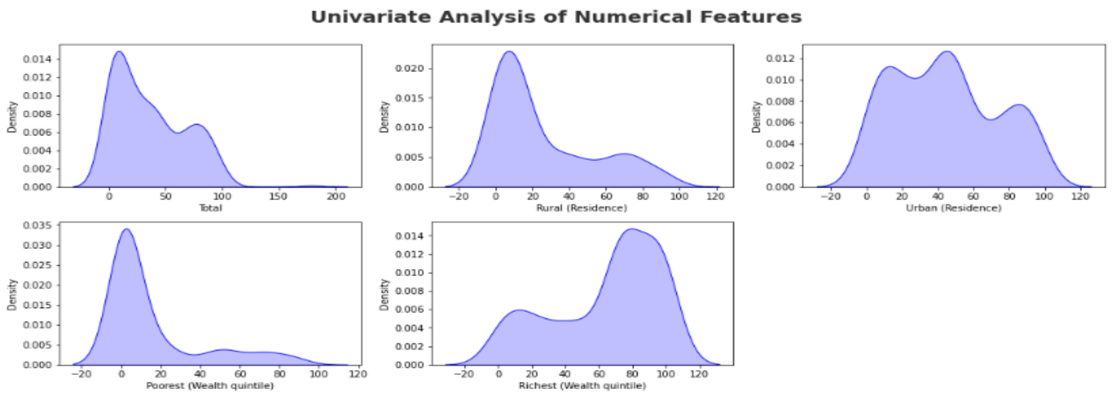
Inference:

1. From the above Boxplot we can see that there is presence of outliers in the DataFrame1.
2. There are some outliers present in the Rural and Poorest we can reduce those by using IQR method but it doesn't make sense.
3. When we do the IQR some records are lost and thus we can proceed further with these Outliers.
4. They all fall in the same scale and there is no need of scaling.

```
In [42]: it=1
         plt.figure(figsize=(11,7))
         for i in df1.select_dtypes(include=np.number):
             plt.subplot(3,2,it)
             sns.histplot(df1[i])
             it=it+1
         plt.tight_layout()
         plt.show()
```



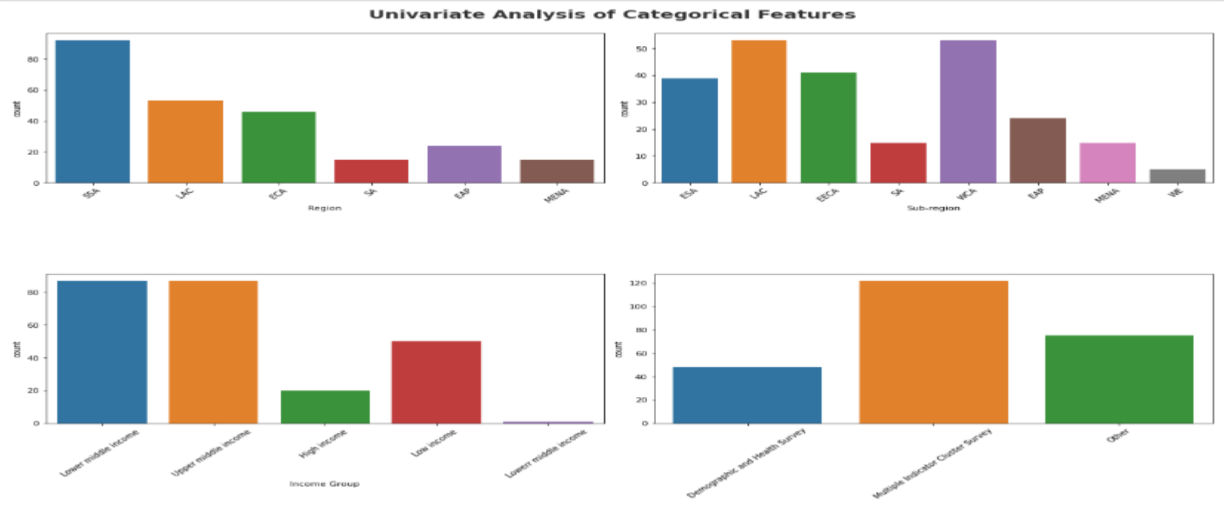
```
In [108]: # define numerical & categorical columns
          numeric_features = [feature for feature in merged_df.columns if merged_df[feature].dtype != 'O']
          categorical_features = [feature for feature in merged_df.columns if merged_df[feature].dtype == 'O']
          plt.figure(figsize=(15, 15))
          plt.suptitle('Univariate Analysis of Numerical Features', fontsize=20, fontweight='bold', alpha=0.8, y=1.)

          for i in range(0, len(numeric_features)):
              plt.subplot(5, 3, i+1)
              sns.kdeplot(x=merged_df[numeric_features[i]],shade=True, color='b')
              plt.xlabel(numeric_features[i])
              plt.tight_layout()
```



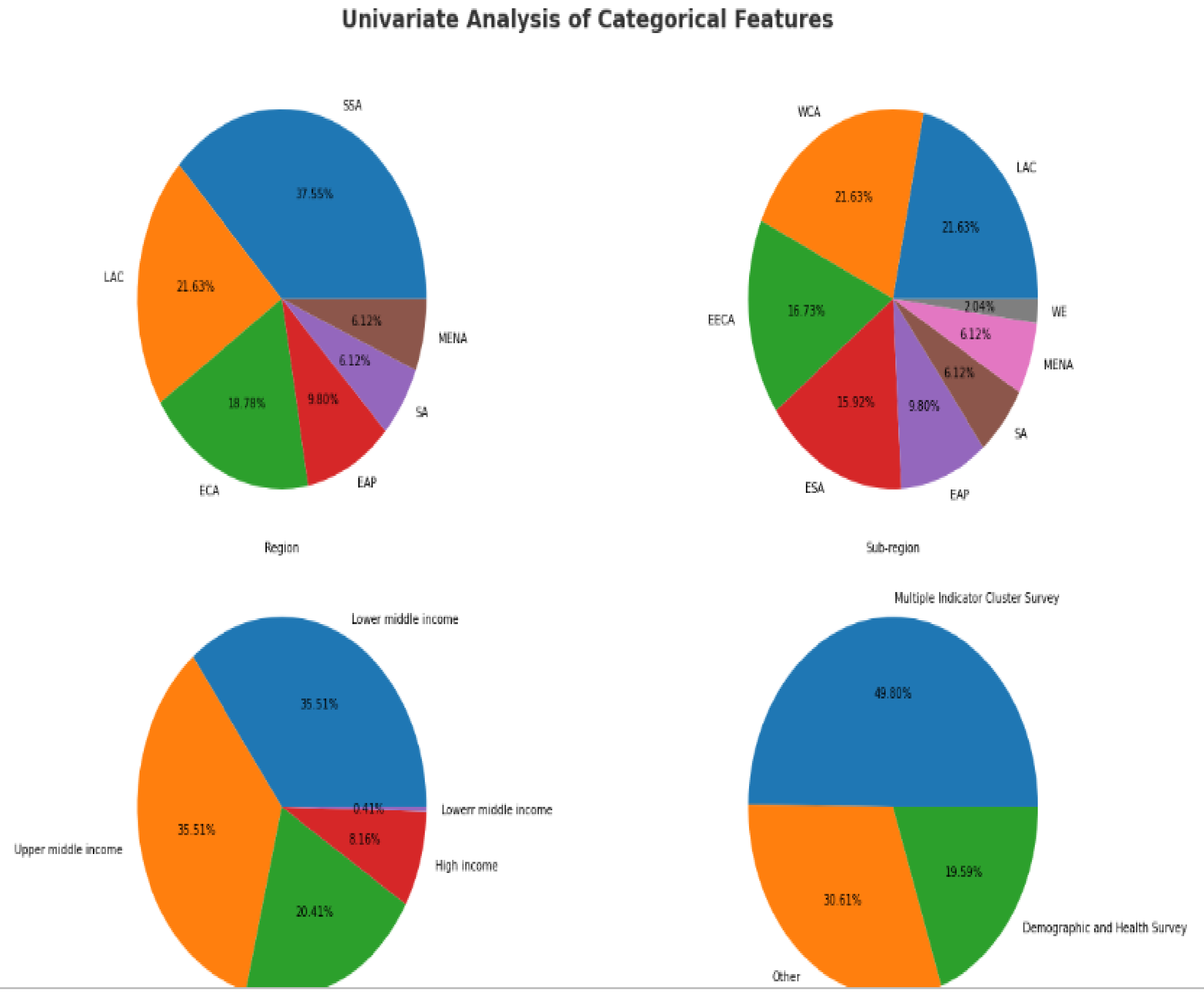
Inference:

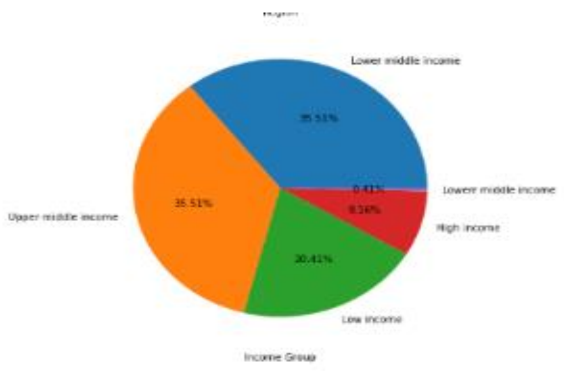
1. Therefore We have done univariate Analysis of Numerical Features.
2. Therefore we can see that Some are right skewed and Richest alone tends to be in Left skew.
3. The merged dataset gives a overall distribution of Numerical variables.

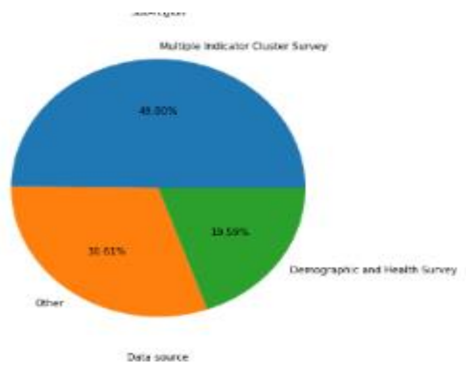
```
In [109]: # categorical columns
          plt.figure(figsize=(17, 27))
          plt.suptitle('Univariate Analysis of Categorical Features', fontsize=20, fontweight='bold', alpha=0.8, y=1.)
          for i in range(0, len(categorical_features)):
              plt.subplot(5, 2, i+1)
              sns.countplot(x=merged_df[categorical_features[i]])
              plt.xlabel(categorical_features[i])
              plt.xticks(rotation=45)
              plt.tight_layout()
```

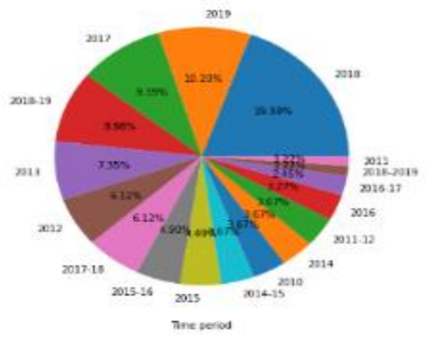


```
In [115]: plt.figure(figsize=(17, 27))
          plt.suptitle('Univariate Analysis of Categorical Features', fontsize=20, fontweight='bold', alpha=0.8, y=1.)
          for i in range(0, len(categorical_features)):
              plt.subplot(5, 2, i+1)
              plt.pie(merged_df[categorical_features[i]].value_counts(),labels=merged_df[categorical_features[i]].value_counts().index,aut
              plt.xlabel(categorical_features[i])
              plt.xticks(rotation=45)
              plt.tight_layout()
```





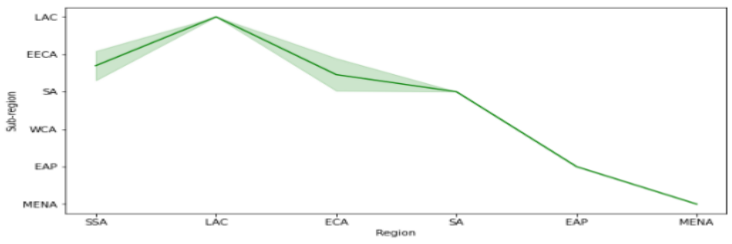




Inference:

1. Therefore we have categorical columns with their percentage of the contribution.
2. This piechart shows us that In Region SSA is more than 37% and In case Income Group we can see that upper Middle Income and Lower Middle Income are similar in their respective Percentage.

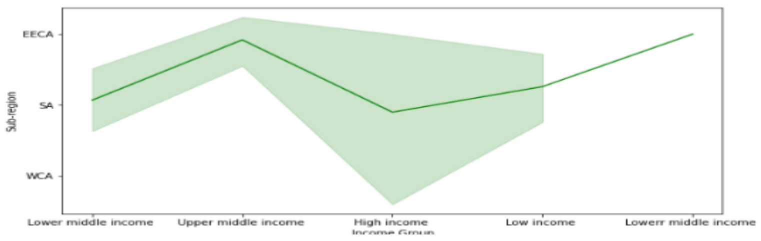
In [110]:
```
plt.subplots(figsize=(10,5))
sns.lineplot(x='Region',y='Sub-region',data=merged_df,color='g')
plt.show()
```



Inference:

1. The plot displays the information about the 2 categorical features that is Region and Sub-Region.
2. This shows that how the region and sub-region are related to each other and therefore we can there is Quite a higher peak in the front and then decrease drastically.

In [111]:
```
plt.subplots(figsize=(10,5))
sns.lineplot(x='Income Group',y='Sub-region',data=merged_df,color='g')
plt.show()
```



Converting this Final dataset into a CSV

In [137]:
```
merged_df.to_csv('final.csv')
```

CONCLUSION

Exploratory Data Analysis (EDA) is a crucial step in any data analysis project. It involves examining and understanding the data to gain insights into its characteristics, such as distribution, skewness, and outliers, among others. In this project, we carried out EDA on a dataset, merged it, produced a final CSV, and performed various data wrangling techniques to clean and transform the data.

To begin with, we used seaborn and matplotlib to visualize the distribution of variables in the dataset. Visualization allows us to identify the shape of the distribution, identify outliers, and check for any patterns that may exist in the data. We also used box plots to check for the presence of outliers and the interquartile range (IQR) to quantify the variability of the data. Based on our visualization and analysis, we identified several variables that were highly skewed, which could affect the accuracy of our analyses.

To address this issue, we used power transformation to normalize the data. Power transformation involves applying mathematical functions to the data to transform it into a more normal distribution. We used the Box-Cox transformation, which is a widely used method for power transformation. This transformation helps to reduce the effect of outliers and to make the data more amenable to statistical analysis.

After transforming the data, we performed various data wrangling techniques, including pivot tables, cross-tabulations, and group-by operations. Pivot tables allow us to summarize the data by aggregating it in various ways, such as computing the mean, median, or sum of variables across different dimensions. Cross-tabulations provide a way to examine the relationship between two or more variables in the dataset. Group-by operations allow us to group the data by a specific variable or set of variables and compute summary statistics for each group.