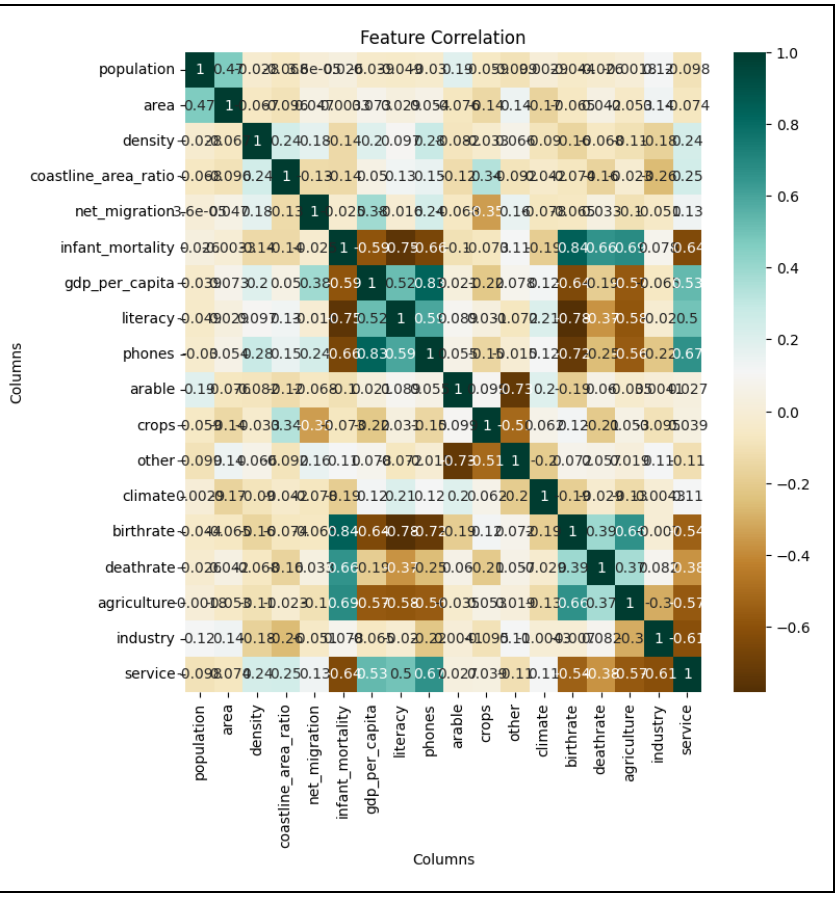
# Data Collection and Preprocessing Phase

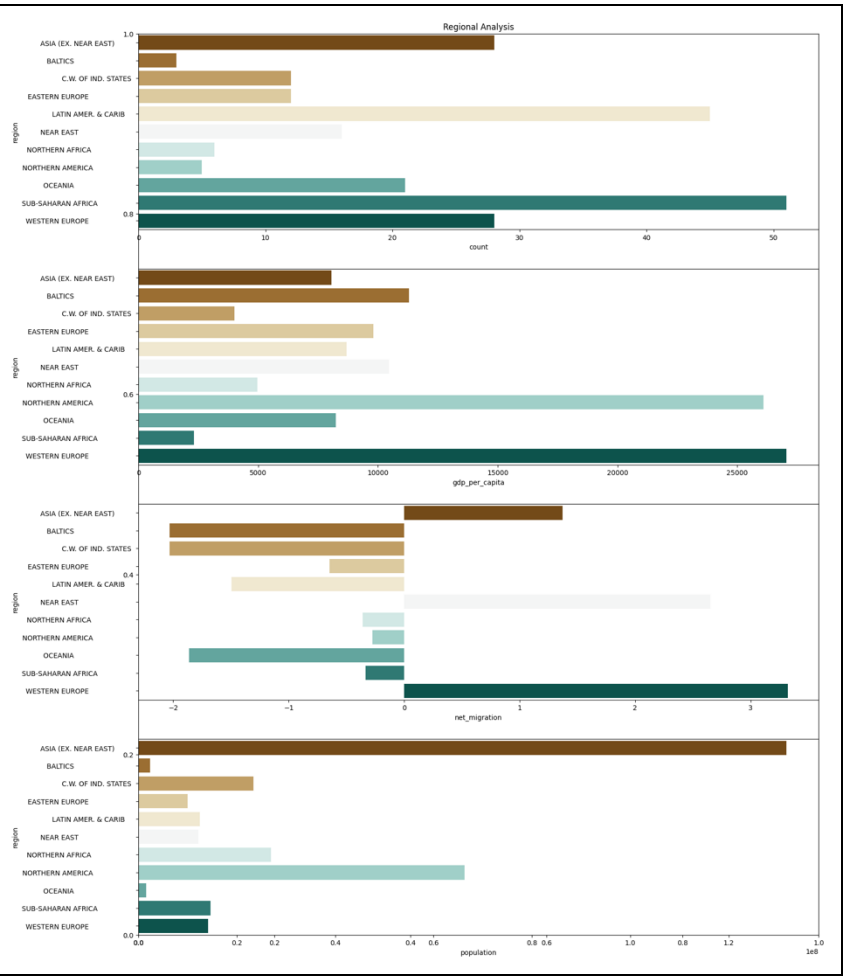
| | |
|---|---|
| Date | 9 JULY 2024 |
| Team ID | 739738 |
| Project Title | Leveraging machine learning for GDP per capita prediction |
| Maximum Marks | 6 Marks |

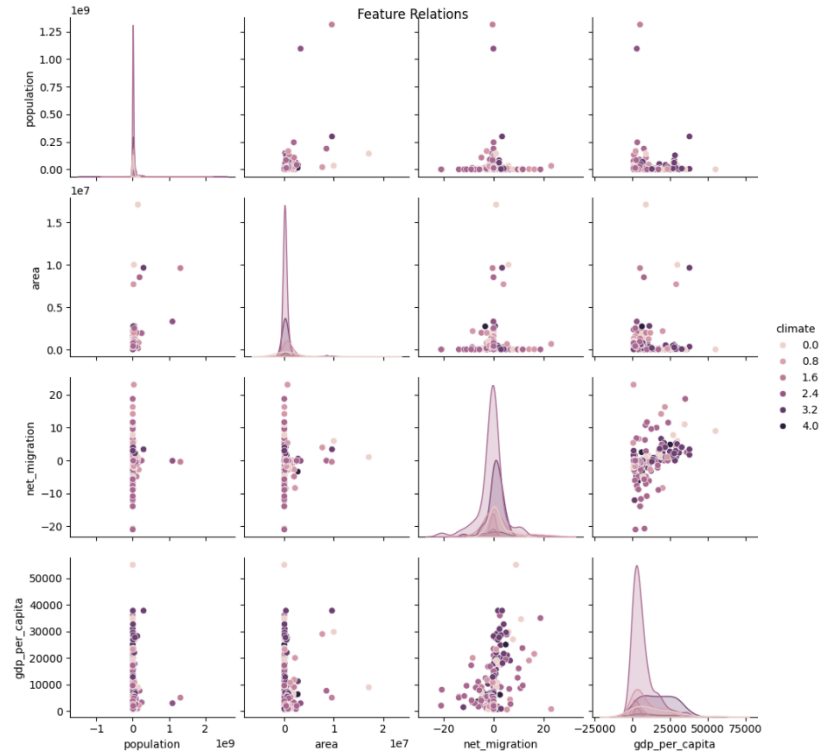
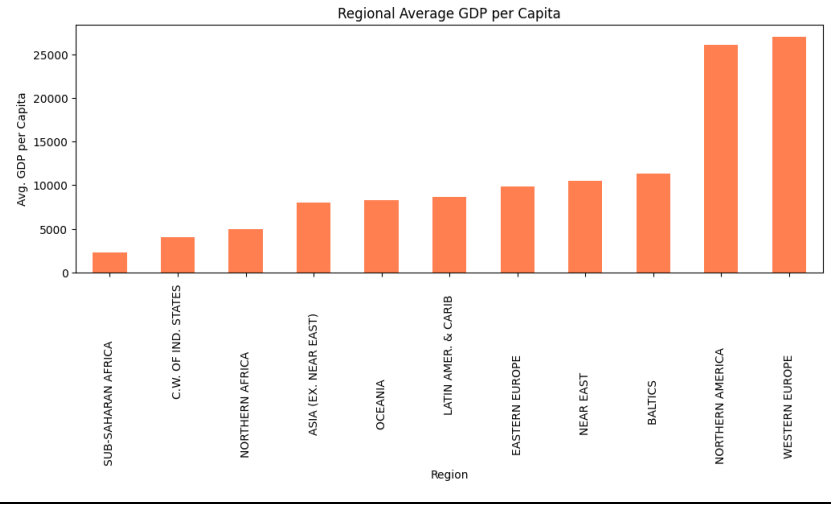
**Preprocessing Template**

The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

| **Section** | **Description** |
|---|---|
| Data Overview | Basics statistics, dimensions and structure of the data. |

| | |
|---|---|
| Bivariate Analysis |  |

| | |
|---|---|
| Bivariate Analysis |  |

| | |
|---|---|
| Multivariate Analysis |  |
| Univariate Analysis |  |
| **Data Preprocessing Code Screenshots** | |

## Loading Data

```
In [ ]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        get_ipython().run_line_magic('matplotlib', 'inline')
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
        from sklearn import metrics
        from sklearn.model_selection import GridSearchCV
        from sklearn.linear_model import LinearRegression
        from sklearn.svm import SVR
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.ensemble import GradientBoostingRegressor
        from sklearn.model_selection import cross_val_score

In [ ]: data = pd.read_csv('/content/countries of the world.csv')

In [ ]: data.head()

Out[ ]:
```

| | Country | Region | Population | Area (sq. mi.) | Pop. Density (per sq. mi.) | Coastline (coast/area ratio) | Net migration | Infant mortality (per 1000 births) | GDP ($ per capita) | Literacy (%) | Phones (per 1000) | Arable (%) | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Afghanistan | ASIA (EX. NEAR EAST) | 31056997 | 647500 | 48,0 | 0,00 | 23,06 | 163,07 | 700.0 | 36,0 | 3,2 | 12,13 | |
| 1 | Albania | EASTERN EUROPE | 3581655 | 28748 | 124,6 | 1,26 | -4,93 | 21,52 | 4500.0 | 86,5 | 71,2 | 21,09 | |
| 2 | Algeria | NORTHERN AFRICA | 32930091 | 2381740 | 13,8 | 0,04 | -0,39 | 31 | 6000.0 | 70,0 | 78,1 | 3,22 | |
| 3 | American Samoa | OCEANIA | 57794 | 199 | 290,4 | 58,29 | -20,71 | 9,27 | 8000.0 | 97,0 | 259,5 | 10 | |

## Handling Missing Data

```
[ ]: print(data.isnull().sum())

country                 0
region                  0
population              0
area                    0
density                 0
coastline_area_ratio    0
net_migration           3
infant_mortality        3
gdp_per_capita          1
literacy               18
phones                  4
arable                  2
crops                   2
other                   2
climate                22
birthrate               3
deathrate               4
agriculture            15
industry               16
service                15
dtype: int64
```

## Data Transformation

```
In [ ]: data['net_migration'].fillna(0, inplace=True)
        data['infant_mortality'].fillna(0, inplace=True)
        data['gdp_per_capita'].fillna(2500, inplace=True)
        data['literacy'].fillna(data.groupby('region')['literacy'].transform('mean'), inplace= True)
        data['phones'].fillna(data.groupby('region')['phones'].transform('mean'), inplace= True)
        data['arable'].fillna(0, inplace=True)
        data['crops'].fillna(0, inplace=True)
        data['other'].fillna(0, inplace=True)
        data['climate'].fillna(0, inplace=True)
        data['birthrate'].fillna(data.groupby('region')['birthrate'].transform('mean'), inplace= True)
        data['deathrate'].fillna(data.groupby('region')['deathrate'].transform('mean'), inplace= True)
        data['agriculture'].fillna(0.17, inplace=True)
        data['service'].fillna(0.8, inplace=True)
        data['industry'].fillna((1 - data['agriculture'] - data['service']), inplace = True)

In [ ]: print(data.isnull().sum())

country                0
region                 0
population             0
area                   0
density                0
coastline_area_ratio   0
net_migration          0
infant_mortality       0
gdp_per_capita         0
literacy               0
phones                 0
arable                 0
crops                  0
other                  0
climate                0
birthrate              0
deathrate              0
agriculture            0
industry               0
service                0
dtype: int64
```