Pronóstico de emisiones de CO2 derivadas del consumo de proteína animal

Alba Sentís - CodeOp Final Project

Según la FAO *"El sector ganadero contribuye significativamente al total de emisiones humanas de gases de efecto invernadero (GEI)"*

- **Selección del dataset**
- **Limpieza de datos**
- **Visualización de datos**
- **Análisis de correlaciones**
- **Modelos de predicción**

Selección del dataset

	Country	Year	Population
0	Afghanistan	1961	9214082
1	Afghanistan	1962	9404411
2	Afghanistan	1963	9604491
3	Afghanistan	1964	9814318
4	Afghanistan	1965	10036003

	Country	Year	Population	Grams_animal_protein	Grams_veg_protein
0	Afghanistan	1961	9214082	13.782217	72.710620
1	Afghanistan	1962	9404411	13.739926	70.840126
2	Afghanistan	1963	9604491	14.483050	64.174194
3	Afghanistan	1964	9814318	14.623308	70.575584
4	Afghanistan	1965	10036003	15.252143	70.359955

	Company	Brief Description	Protein Category	B2B/B2C	Company Focus	Company type	Technology Focus	Product Type	Animal-Type Analog	Ingredient Ty
0	[Mock]	UK-based producer of chef-quality, whole-cut, ...	Plant-based	B2C	Meat	Specialized (focused on alternative proteins)	End product formulation and manufacturing	Whole muscle meat/seafood	Beef/veal,Mutton/lamb,Chicken	Na
1	1Ness Foods Pvt Ltd	India-based producer of plant-based dairy alte...	Plant-based	B2C,B2B	Dairy	Specialized (focused on alternative proteins)	End product formulation and manufacturing,Ingr...	Milk,Cheese,Other	NaN	Cashew,Almond,P
		U.K.-based								

Dataset principal

	Entity	Code	Year	Fish and seafood | 00002960 || Food available for consumption | 0674pc || grams of protein per day per capita	Meat, poultry | 00002734 || Food available for consumption | 0674pc || grams of protein per day per capita	Meat, pig | 00002733 || Food available for consumption | 0674pc || grams of protein per day per capita	Meat, beef | 00002731 || Food available for consumption | 0674pc || grams of protein per day per capita	Meat, sheep and goat | 00002732 || Food available for consumption | 0674pc || grams of protein per day per capita	Meat, Other | 00002735 || Food available for consumption | 0674pc || grams of protein per day per capita	All egg products | 00002744 || Food available for consumption | 0674pc || grams of protein per day per capita	Milk - Excluding Butter | 00002848 || Food available for consumption | 0674pc || grams of protein per day per capita
0	Afghanistan	AFG	1961	0.010186	0.224101	NaN	2.027096	3.167975	0.366711	0.285220	6.346136
1	Afghanistan	AFG	1962	0.010193	0.234435	NaN	2.109914	3.068040	0.377134	0.305785	6.268587
2	Afghanistan	AFG	1963	0.010199	0.234585	NaN	2.131660	3.131195	0.458970	0.305980	6.813153
3	Afghanistan	AFG	1964	0.010205	0.244912	NaN	2.122574	3.224680	0.438801	0.316345	6.857546
4	Afghanistan	AFG	1965	0.010209	0.255223	NaN	2.103040	3.338321	0.469611	0.326686	7.319804

Maddison Project Database 2023

The Maddison Project Database provides information on comparative economic growth and income levels over the very long run. The 2023 version of this database covers 169 countries and the period up to 2022. The new estimates are presented and discussed in Bolt and Van Zanden (2024), "Maddison style estimates of the evolution of the world economy: A new 2023 update", Journal of Economic Surveys, 1–41. For questions not covered in the documentation, please contact ggdc@rug.nl .

	Country ISO code	Country name	Regional grouping	Country class
0	ABW	Aruba	Rest Central America	Developing
1	AFG	Afghanistan	India +	Developing
2	AGO	Angola	Southern_Africa	Developing
3	AIA	Anguilla	Rest Central America	Developing
4	AIR	Int. Aviation	Int. Aviation	0

	Code	Country	Region	Year	Gdp_pc_$	Population Thousands
0	AFG	Afghanistan	South and South East Asia	1961	1309.0	10043.0
1	AFG	Afghanistan	South and South East Asia	1962	1302.0	10267.0
2	AFG	Afghanistan	South and South East Asia	1963	1298.0	10501.0
3	AFG	Afghanistan	South and South East Asia	1964	1291.0	10744.0
4	AFG	Afghanistan	South and South East Asia	1965	1290.0	10998.0

	Food Type	$kgCO_2$_p100gr
0	Apples	14.333333
1	Bananas	9.555556
2	Beef (beef herd)	49.889669
3	Beef (dairy herd)	16.869301
4	Berries & Grapes	15.300000

Dificultades

- Nombres de los países
- Países que ya no existen o no existían en algunas fechas
- Gestión de los valores faltantes

```
['Africa',
 'Africa (FAO)',
 'Americas (FAO)',
 'Asia',
 'Asia (FAO)',
 'Belgium-Luxembourg (FAO)',
 'Caribbean (FAO)',
 'Central America (FAO)',
 'Central Asia (FAO)',
 'China (FAO)',
 'Eastern Africa (FAO)',
 'Eastern Asia (FAO)',
 'Eastern Europe (FAO)',
 'Europe',
 'Europe (FAO)',
 'European Union (27)',
 'European Union (27) (FAO)',
 'High-income countries',
 'Land Locked Developing Countries (FAO)',
 'Least Developed Countries (FAO)',
 'Low Income Food Deficit Countries (FAO)',
 'Low-income countries',
 'Lower-middle-income countries',
 'Micronesia (FAO)',
 'Middle Africa (FAO)',
 'Net Food Importing Developing Countries (FAO)',
 'North America',
 'Northern Africa (FAO)',
 'Northern America (FAO)',
```

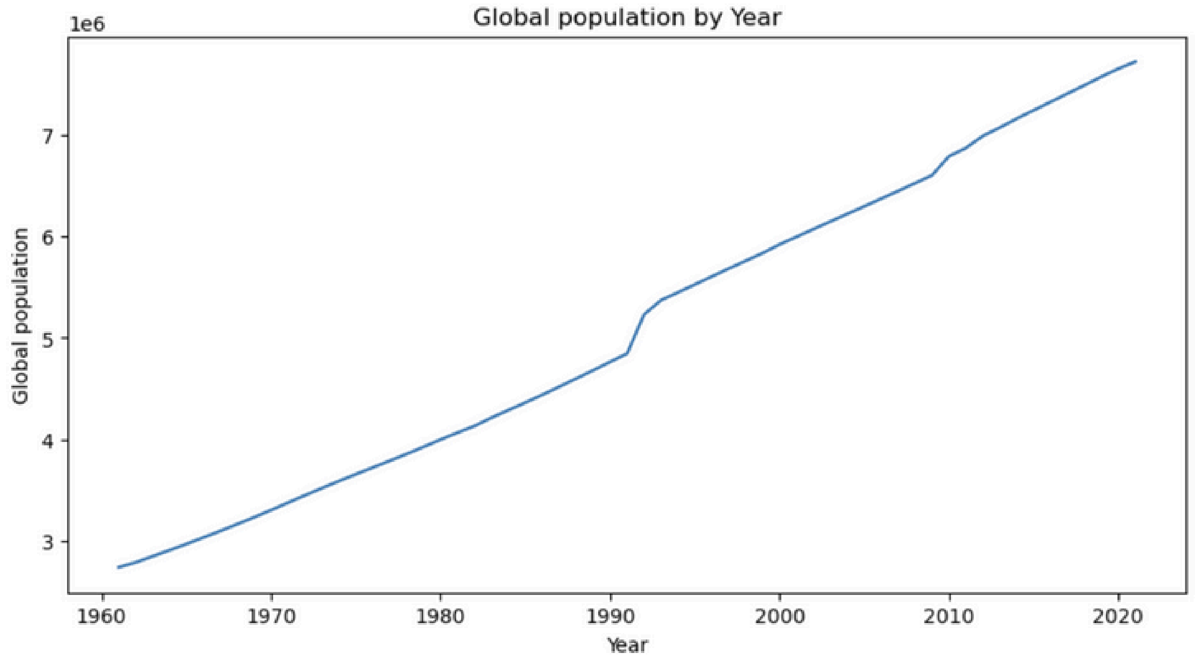
```
Code                             0
Country_x                        0
Region                           0
Year                             0
Gdp_pc_$                         0
Population Thousands             0
Country Class                 1971
Country_y                     1971
Fish & Seafood                1971
Poultry Meat                  1971
Pig Meat                      1971
Beef Meat                     1971
Lamb & Mutton                 1971
Other Meat                    1971
Eggs                          1971
Dairy                         1971
Total Protein (capita/day)    1971
dtype: int64
```

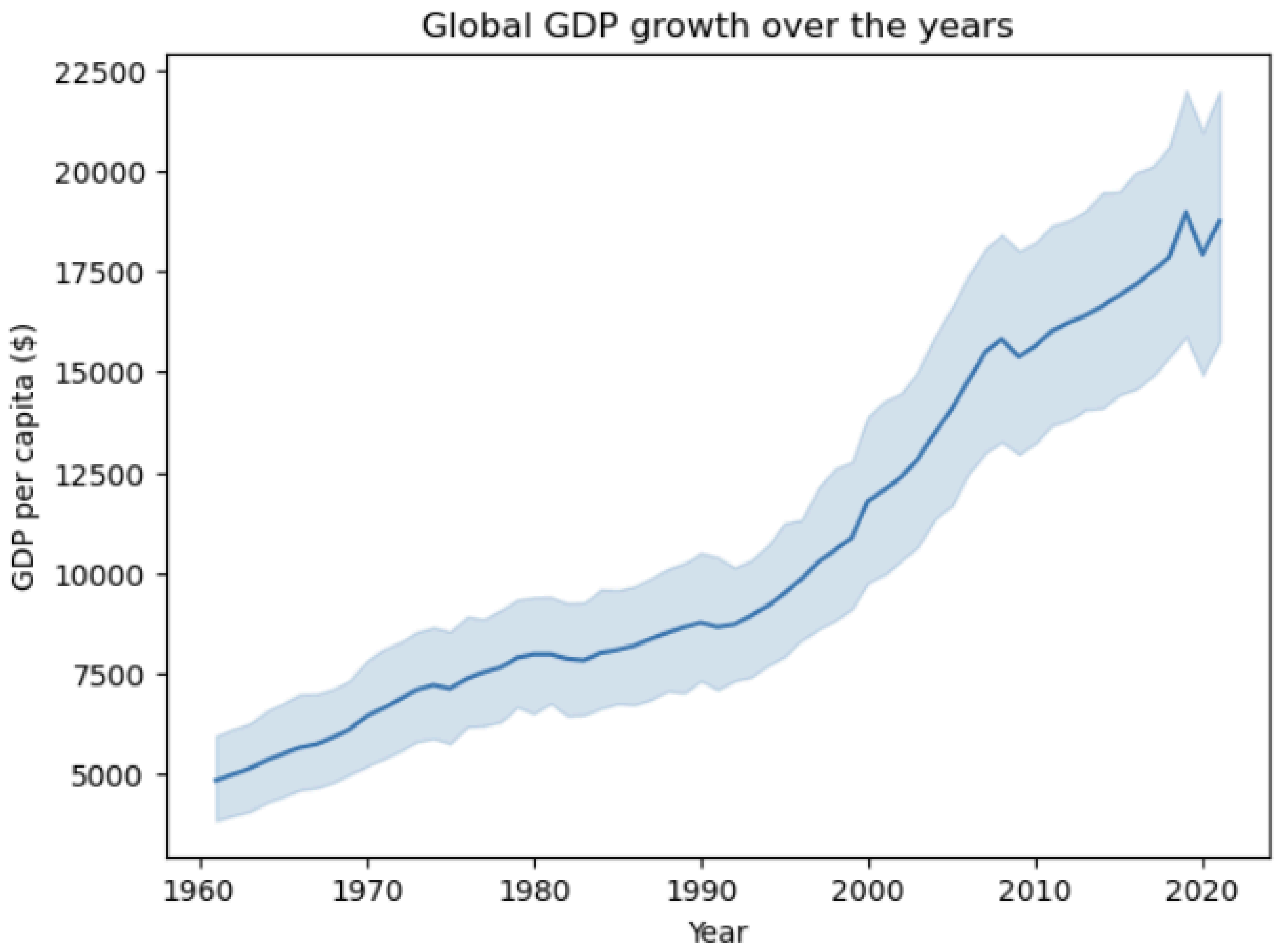
```python
def ussr_gdp(df, list):
    """Takes the NaN rows of the dataframe gdp column and replaces it for the USSR value if it's in the Eastern Countries list """
    ussr_gdp_dict = df[df["Country"] == "Former USSR"][["Year", "Gdp_pc_$"]].set_index("Year").to_dict()["Gdp_pc_$"]
    excluded_countries = ["United Arab Emirates"]
    for i, row in df.iterrows():
        if row["Country"] in list and pd.isna(row["Gdp_pc_$"]):
            year = row["Year"]
            if year in ussr_gdp_dict:
                df.at[i, "Gdp_pc_$"] = ussr_gdp_dict[year]
    return df

new_df = ussr_gdp(maddison_project_df, eastern_e
new_df
```

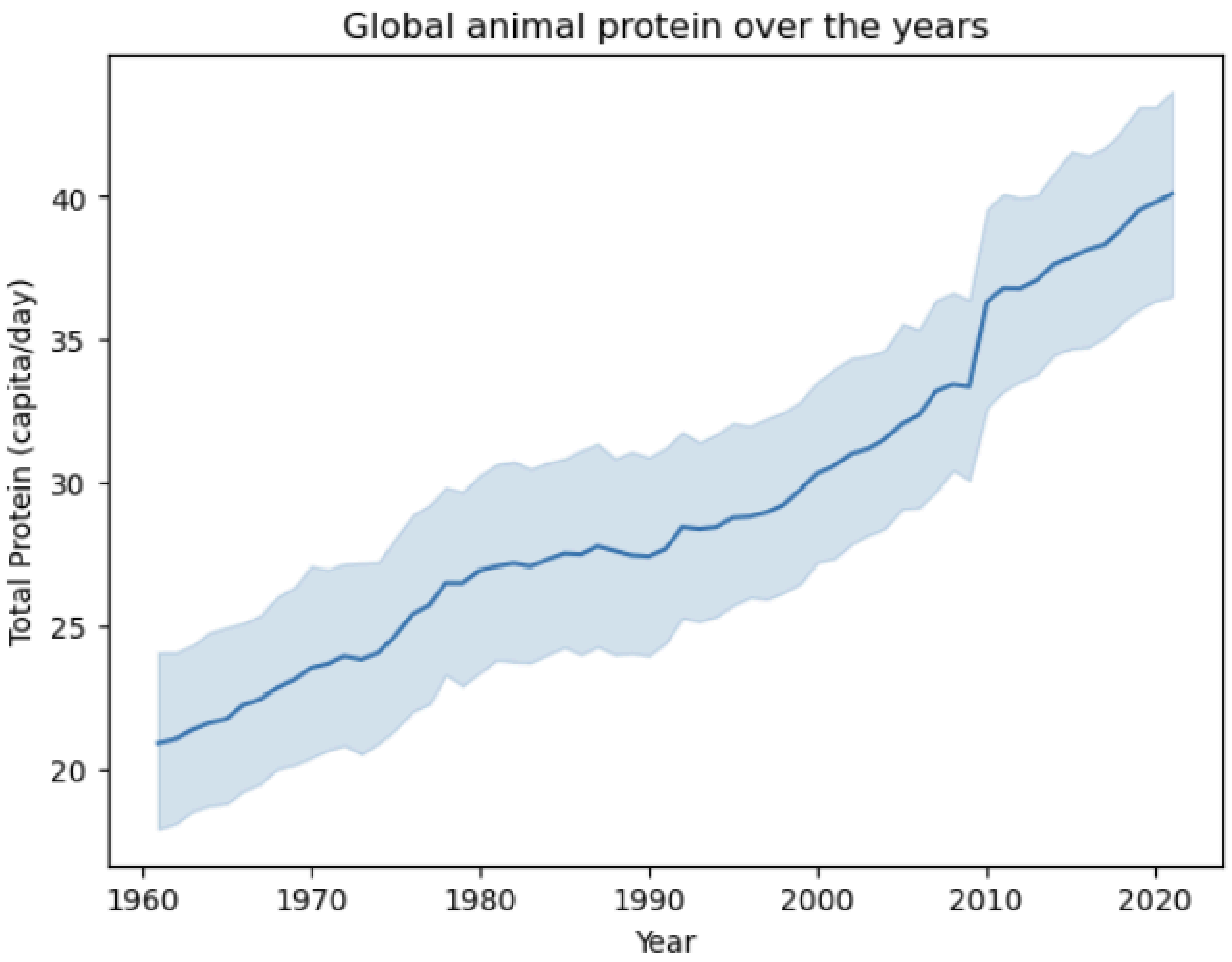
```python
#Create list of country names
list_maddison = sorted(maddison_project_df["Country"].unique().tolist())
list_animals = sorted(animal_protein_df["Country"].unique().tolist())

#Define a function that finds country names that are similar but not exactly the same
def similar_names(list1, list2, length):
    """ Finds country names in two lists that have similar names but are not exactly the same.
    Length is the minimum length of the matching substring."""

    def has_common_substring(str1, str2, length):
        for i in range(len(str1) - length + 1):
            if str1[i:i + length] in str2:
                return True
        return False

    similar_countries = []
    for country1 in list1:
        for country2 in list2:
            if country1 != country2 and has_common_substring(country1, country2, length):
                similar_countries.append((country1, country2))

    return similar_countries
```

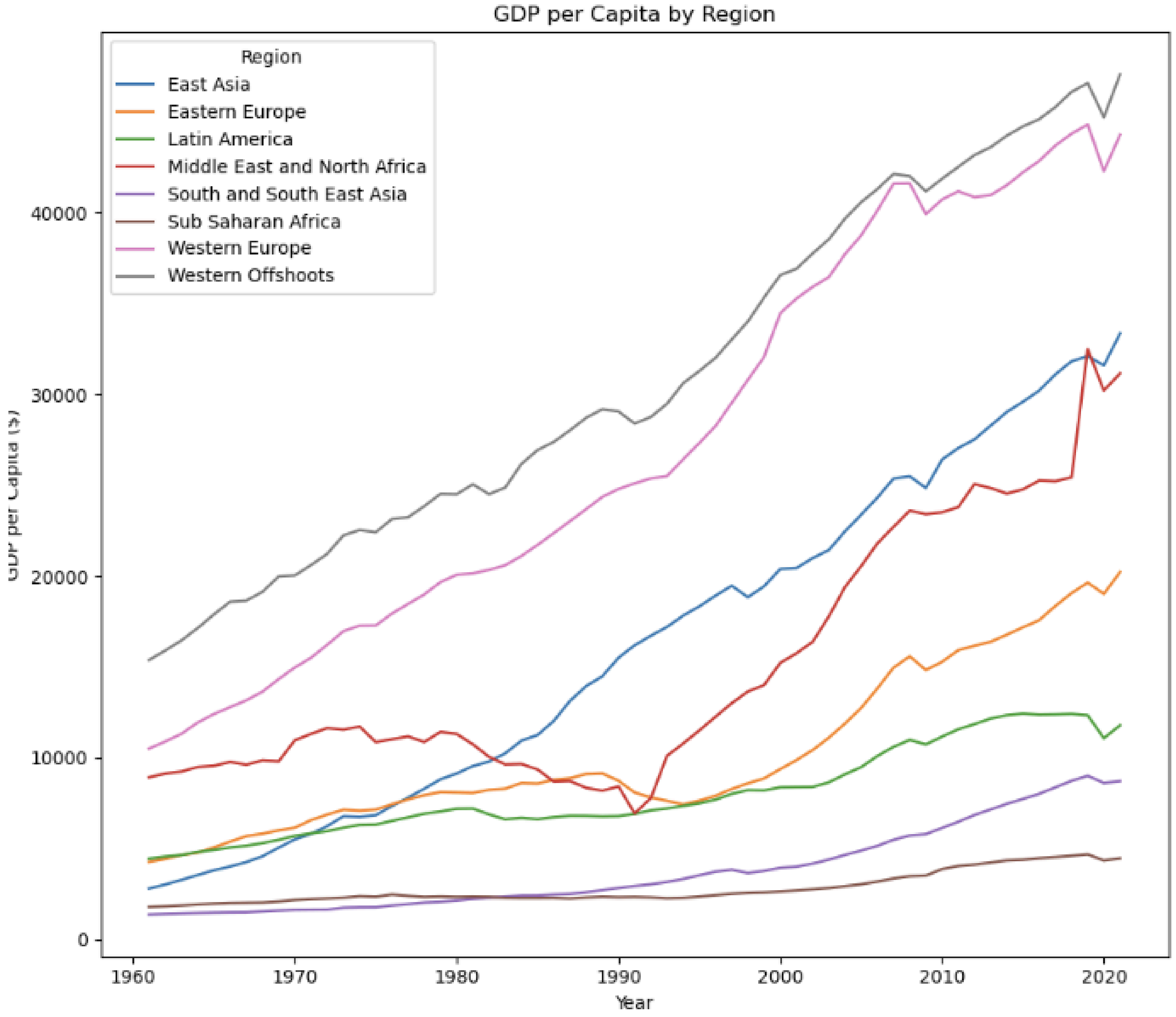
```
'Bolivia (Plurinational State of)',
'Brazil',
'Barbados',
'Botswana',
'Central African Republic',
'Canada',
'Switzerland',
'Chile',
'China',
"Côte d'Ivoire",
'Cameroon',
'D.R. of the Congo',
'Congo',
```

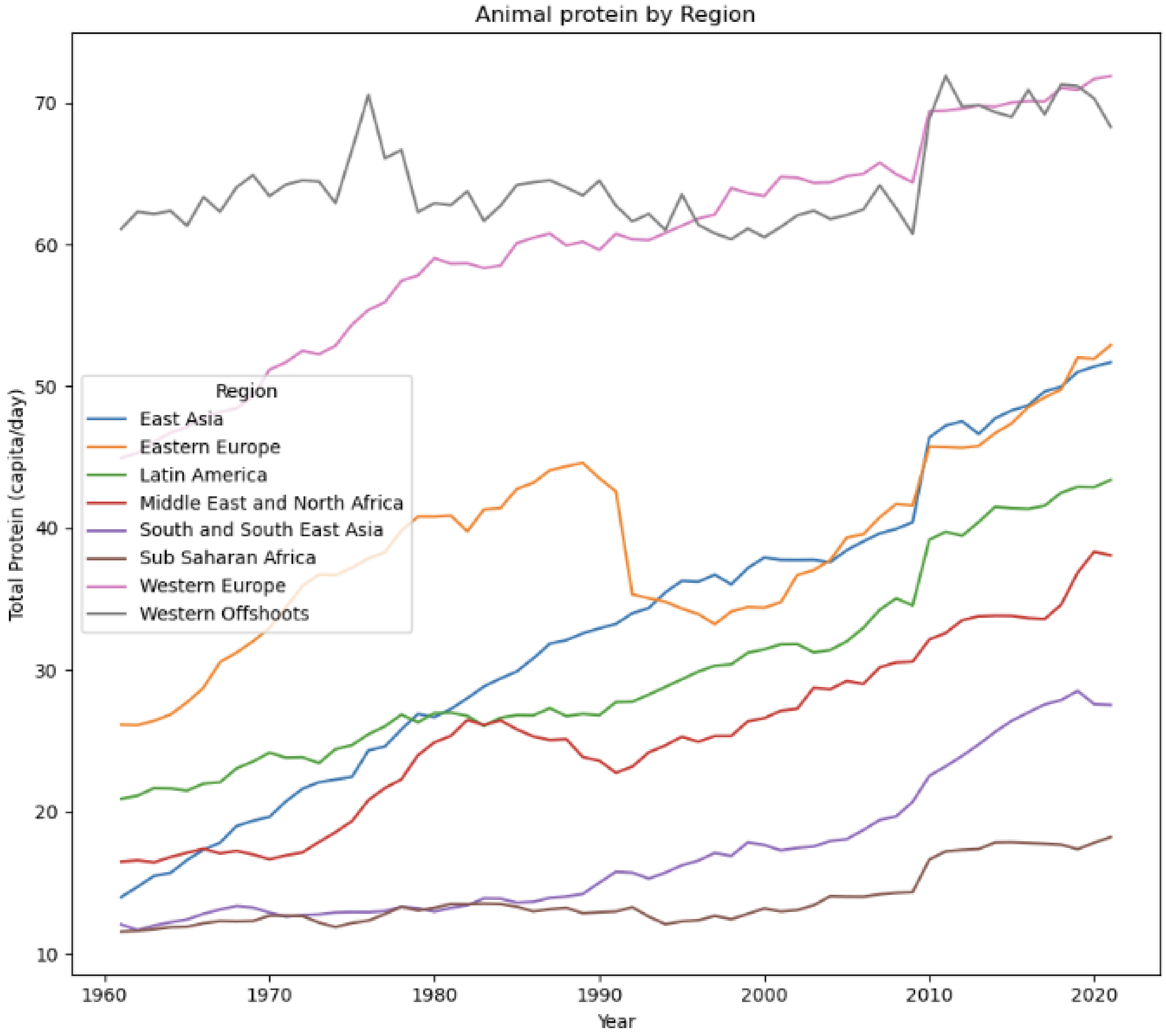
Visualización de datos

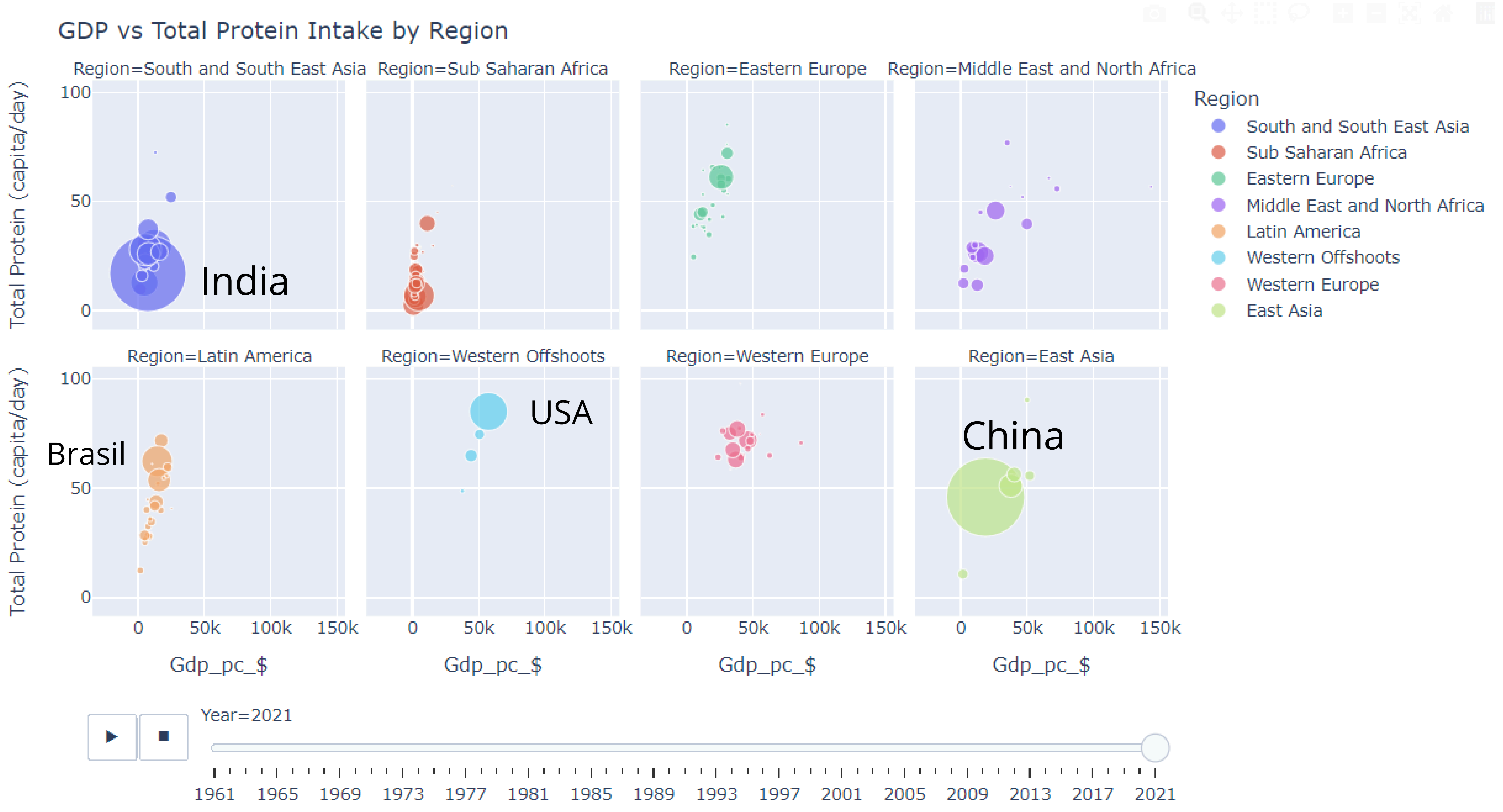




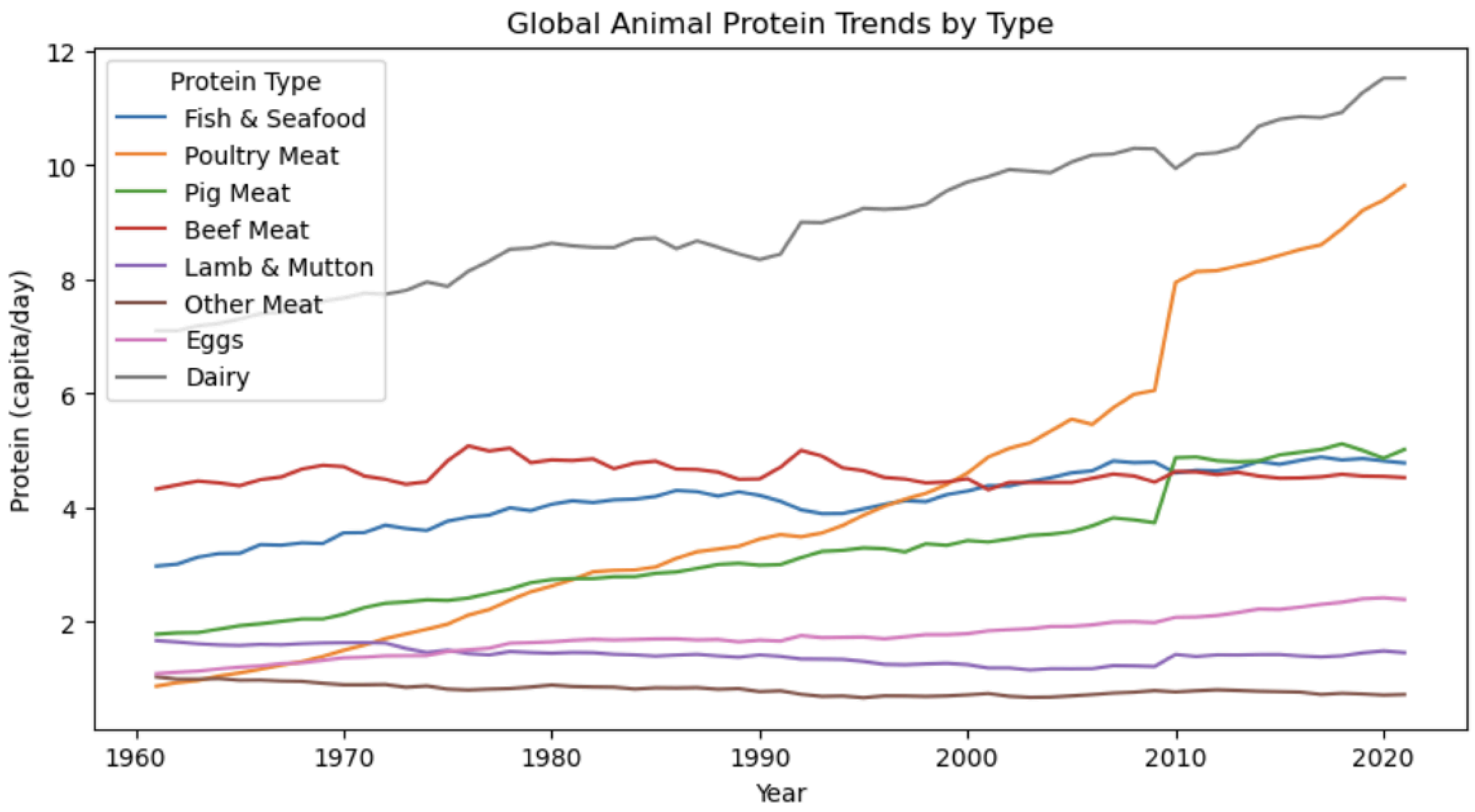



GDP per Capita by Region
Region
East Asia
Eastern Europe
Latin America
Middle East and North Africa
South and South East Asia
Sub Saharan Africa
Western Europe
Western Offshoots
40000
30000
20000
10000
0
1960
1970
1980
1990
2000
2010
2020
Year

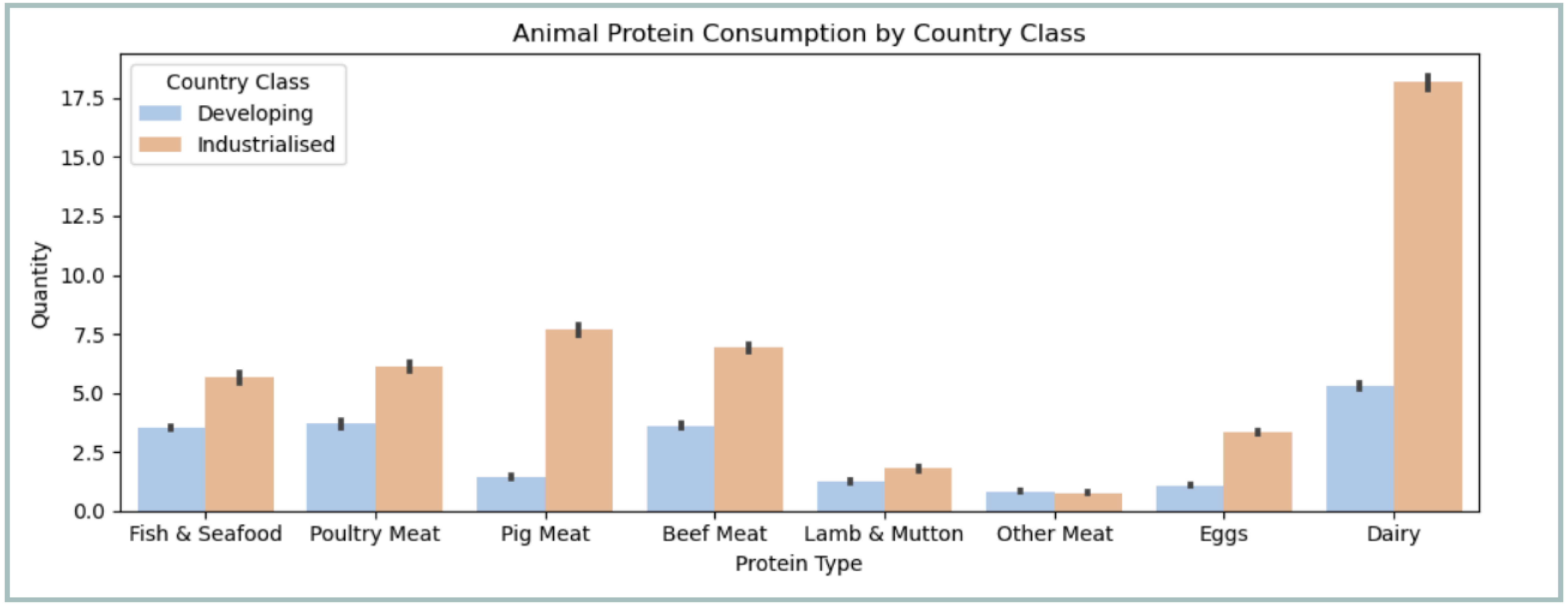

Animal protein by Region
Region
East Asia
Eastern Europe
Latin America
Middle East and North Africa
South and South East Asia
Sub Saharan Africa
Western Europe
Western Offshoots
Total Protein (capita/day)
70
60
50
40
30
20
10
1960
1970
1980
1990
2000
2010
2020
Year

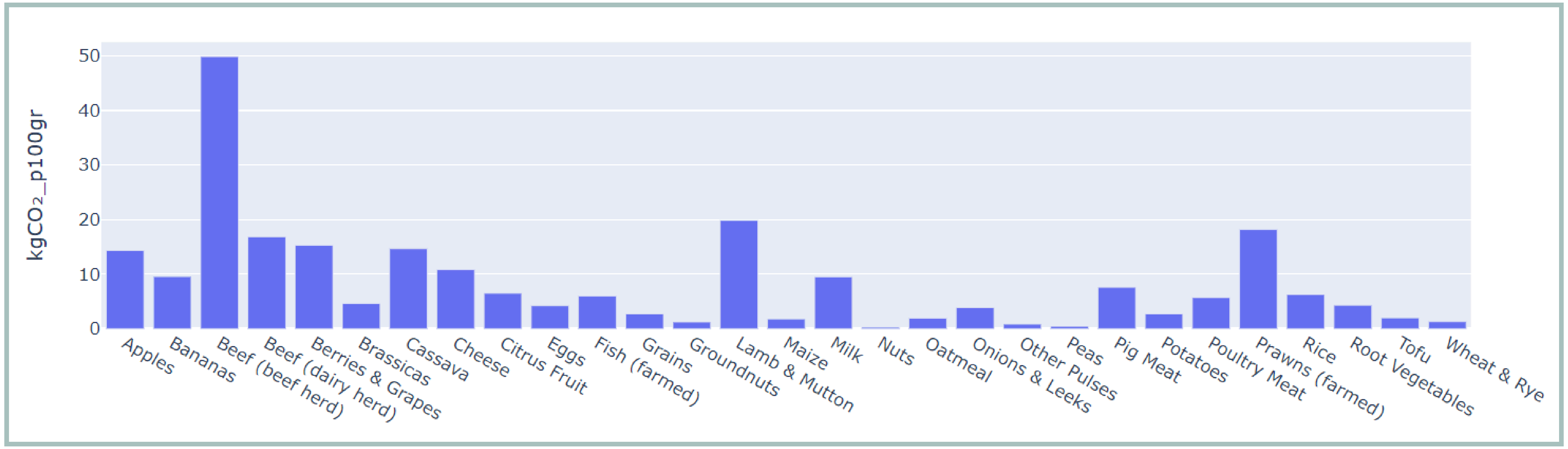


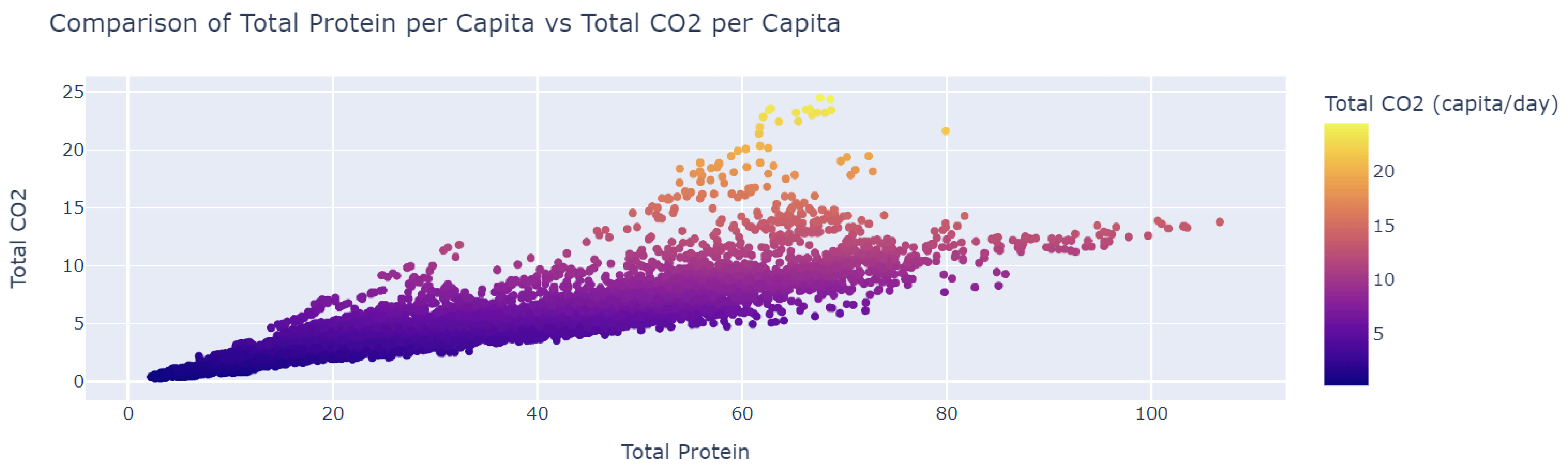
GDP vs Total Protein Intake by Region
Region=South and South East Asia
Region=Sub Saharan Africa
Region=Eastern Europe
Region=Middle East and North Africa
Region=Latin America
Region=Western Offshoots
Region=Western Europe
Region=East Asia
Total Protein (capita/day)
100
50
0
0
50k
100k
150k
Gdp_pc_$
India
Brasil
USA
China
Region
South and South East Asia
Sub Saharan Africa
Eastern Europe
Middle East and North Africa
Latin America
Western Offshoots
Western Europe
East Asia
Year=2021
1961
1965
1969
1973
1977
1981
1985
1989
1993
1997
2001
2005
2009
2013
2017
2021

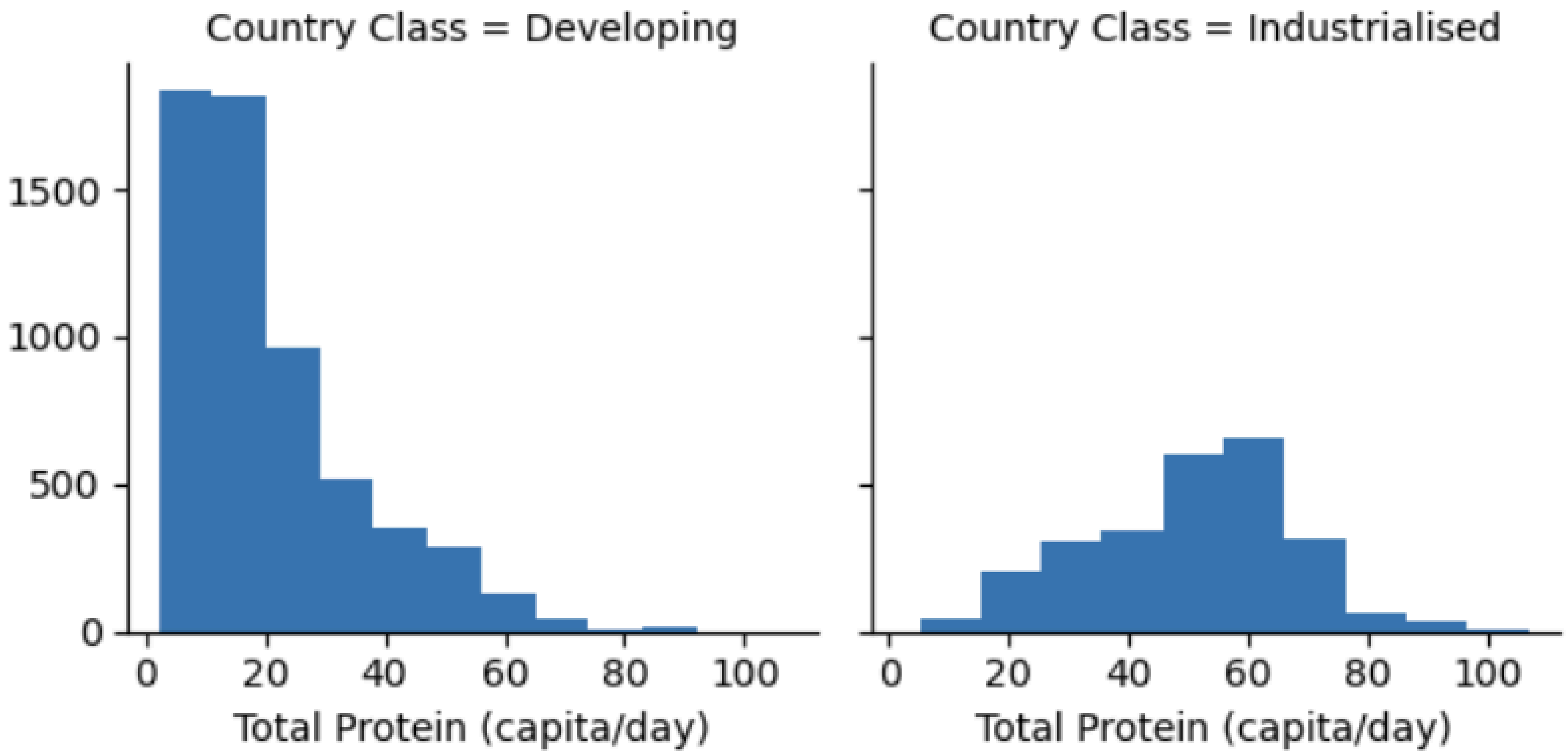


Global Animal Protein Trends by Type
Protein Type
Fish & Seafood
Poultry Meat
Pig Meat
Beef Meat
Lamb & Mutton
Other Meat
Eggs
Dairy
Protein (capita/day)
Year
12
10
8
6
4
2
1960
1970
1980
1990
2000
2010
2020


Animal Protein Consumption by Country Class
Country Class
Developing
Industrialised
Quantity
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Fish & Seafood
Poultry Meat
Pig Meat
Beef Meat
Lamb & Mutton
Other Meat
Eggs
Dairy
Protein Type

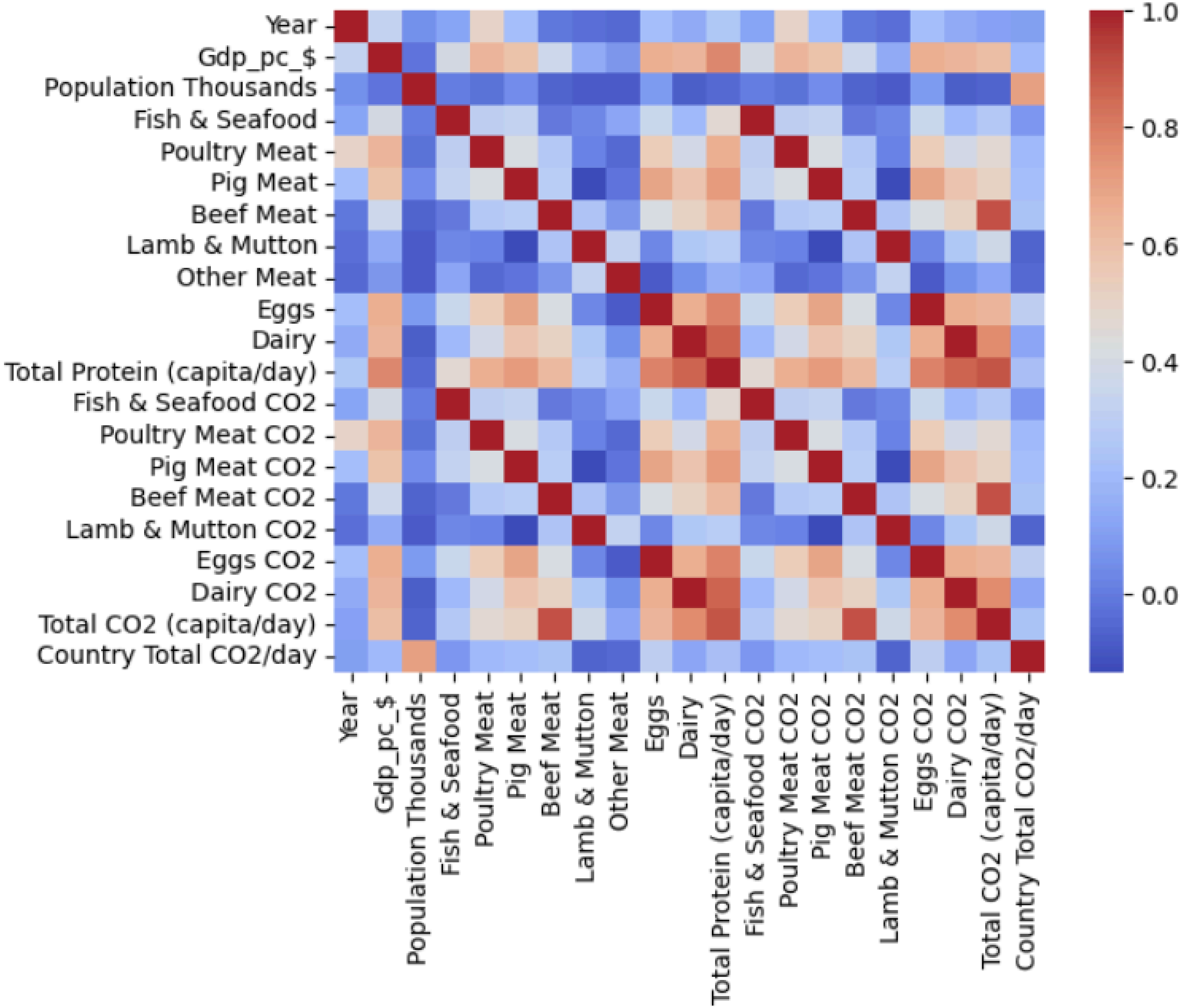

kgCO₂_p100gr
50
40
30
20
10
0
Apples
Bananas
Beef (beef herd)
Beef (dairy herd)
Berries & Grapes
Brassicas
Cassava
Cheese
Citrus Fruit
Eggs
Fish (farmed)
Grains
Groundnuts
Lamb & Mutton
Maize
Milk
Nuts
Oatmeal
Onions & Leeks
Other Pulses
Peas
Pig Meat
Potatoes
Poultry Meat
Prawns (farmed)
Rice
Root Vegetables
Tofu
Wheat & Rye


Comparison of Total Protein per Capita vs Total CO2 per Capita
Total CO2
25
20
15
10
5
0
0
20
40
60
80
100
Total Protein
Total CO2 (capita/day)
20
15
10
5



Country Class = Developing
Country Class = Industrialised
1500
1000
500
0
0
20
40
60
80
100
Total Protein (capita/day)
Total Protein (capita/day)

<Axes: >

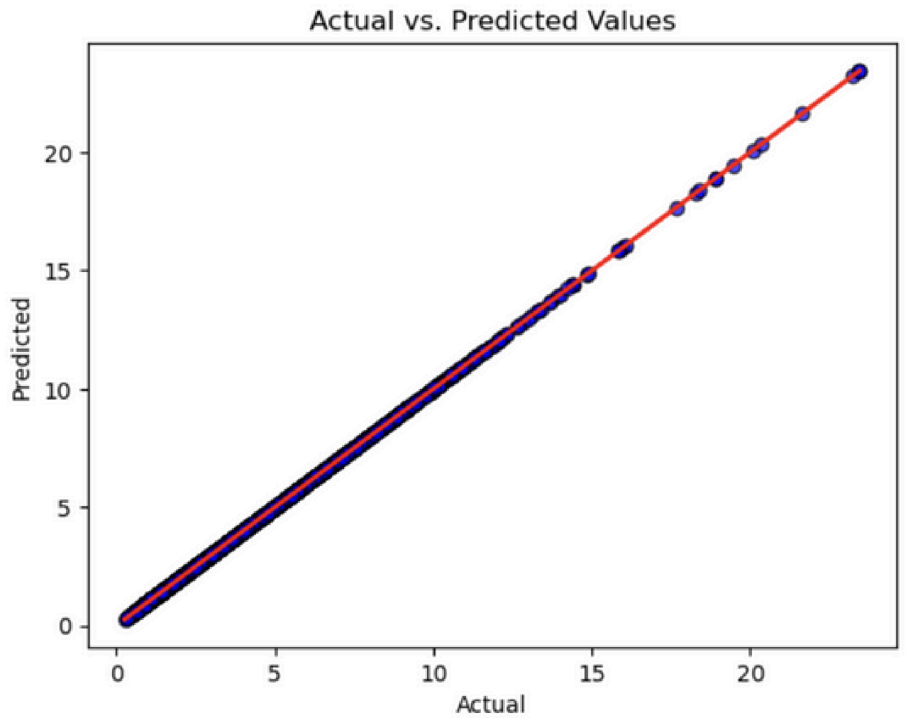

Year
Gdp_pc_$
Population Thousands
Fish & Seafood
Poultry Meat
Pig Meat
Beef Meat
Lamb & Mutton
Other Meat
Eggs
Dairy
Total Protein (capita/day)
Fish & Seafood CO2
Poultry Meat CO2
Pig Meat CO2
Beef Meat CO2
Lamb & Mutton CO2
Eggs CO2
Dairy CO2
Total CO2 (capita/day)
Country Total CO2/day
1.0
0.8
0.6
0.4
0.2
0.0

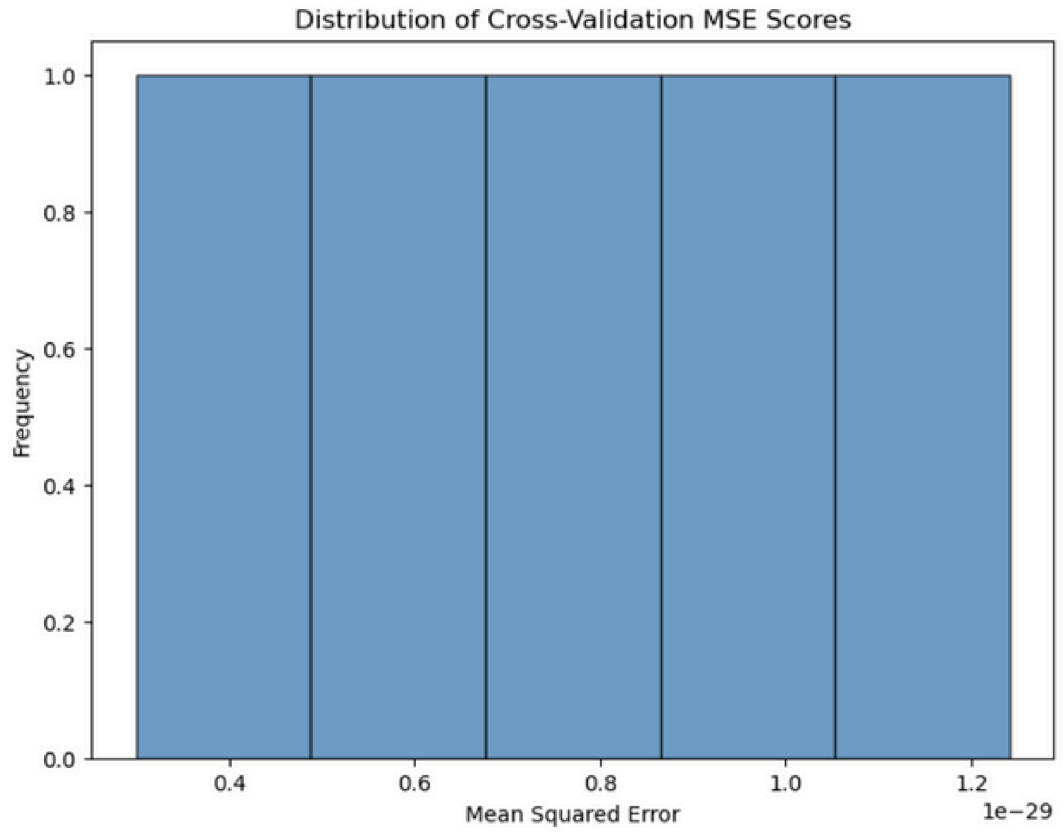
Selección de características

Gdp_pc_$	Population Thousands	Country Class	Fish & Seafood	Poultry Meat	Pig Meat	Beef Meat	Lamb & Mutton	Other Meat	Eggs	...	Total Protein (capita/day)	Total CO2 (capita/day)	Region_East Asia	Region_Eastern Europe	Region_L Ame
1309.0	10043.0	0	0.010186	0.224101	0.0	2.027096	3.167975	0.366711	0.285220	...	12.427424	2.310778	0	0	
1302.0	10267.0	0	0.010193	0.234435	0.0	2.109914	3.068040	0.377134	0.305785	...	12.374089	2.325837	0	0	
1298.0	10501.0	0	0.010199	0.234585	0.0	2.131660	3.131195	0.458970	0.305980	...	13.085742	2.404555	0	0	
1291.0	10744.0	0	0.010205	0.244912	0.0	2.122574	3.224680	0.438801	0.316345	...	13.215063	2.424113	0	0	
1290.0	10998.0	0	0.010209	0.255223	0.0	2.103040	3.338321	0.469611	0.326686	...	13.822893	2.484903	0	0	

Region_South and South East Asia	Region_Sub Saharan Africa	Region_Western Europe	...	Dairy	Total Protein (capita/day)	Fish & Seafood CO2	Poultry Meat CO2	Pig Meat CO2	Beef Meat CO2	Lamb & Mutton CO2	Eggs CO2	Dairy CO2	Total CO2 (capita/day)
1	0	0	...	0.150364	0.097473	0.000282	0.006385	0.000000	0.044969	0.101635	0.035967	0.150364	0.084463
1	0	0	...	0.148524	0.096963	0.000282	0.006679	0.000000	0.046834	0.098429	0.038561	0.148524	0.085085
1	0	0	...	0.161446	0.103772	0.000282	0.006683	0.000000	0.047324	0.100455	0.038585	0.161446	0.088336
1	0	0	...	0.162500	0.105009	0.000282	0.006978	0.000000	0.047119	0.103455	0.039892	0.162500	0.089143
1	0	0	...	0.173469	0.110824	0.000282	0.007271	0.000000	0.046679	0.107100	0.041196	0.173469	0.091654
...	...	...	...	...	...	...	...	...	...	...	...	...	...

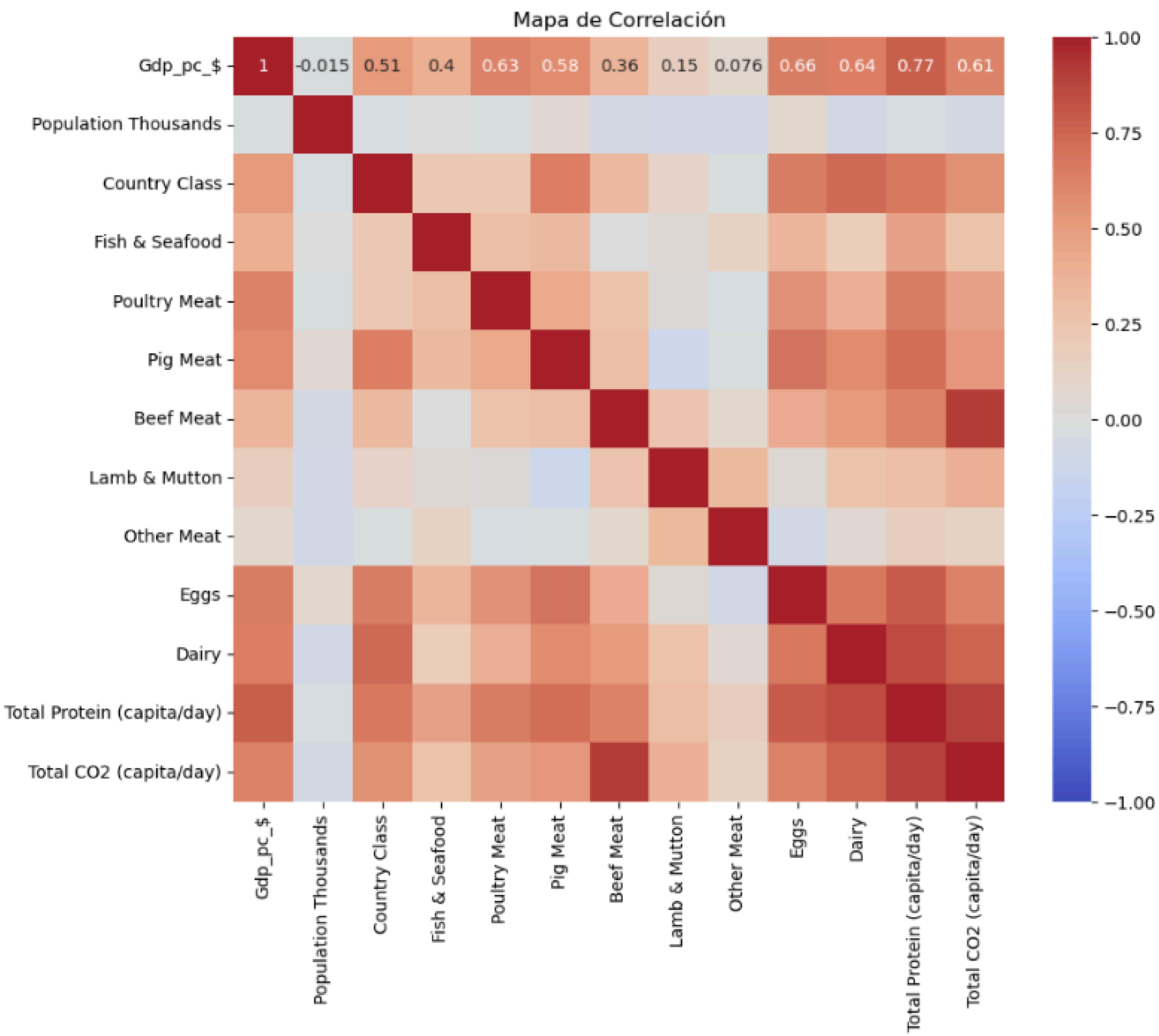
```python
# Ordenar los datos por año
merged_df = merged_df.sort_values(by='Year')

# Determinar el punto de corte para el 80% de los años
unique_years = merged_df['Year'].unique()
cutoff_year = unique_years[int(len(unique_years) * 0.8)]

# Crear conjuntos de entrenamiento y prueba basados en el año
train_df = merged_df[merged_df['Year'] <= cutoff_year]
test_df = merged_df[merged_df['Year'] > cutoff_year]
```






```python
from sklearn.model_selection import TimeSeriesSplit, cross_val_score

# Usar TimeSeriesSplit para respetar la secuencia temporal
tscv = TimeSeriesSplit(n_splits=5)
cross_val_scores = cross_val_score(model, X, y, cv=tscv, scoring='r2')
```

	Gdp_pc_$	Population Thousands	Country Class	Fish & Seafood	Poultry Meat	Pig Meat	Beef Meat	Lamb & Mutton	Other Meat	Eggs	Dairy	Total Protein (capita/day)	Total CO2 (capita/day)
0	1309.0	10043.0	0	0.010186	0.224101	0.0	2.027096	3.167975	0.366711	0.285220	6.346136	12.427424	2.310778
1	1302.0	10267.0	0	0.010193	0.234435	0.0	2.109914	3.068040	0.377134	0.305785	6.268587	12.374089	2.325837
2	1298.0	10501.0	0	0.010199	0.234585	0.0	2.131660	3.131195	0.458970	0.305980	6.813153	13.085742	2.404555
3	1291.0	10744.0	0	0.010205	0.244912	0.0	2.122574	3.224680	0.438801	0.316345	6.857546	13.215063	2.424113
4	1290.0	10998.0	0	0.010209	0.255223	0.0	2.103040	3.338321	0.469611	0.326686	7.319804	13.822893	2.484903

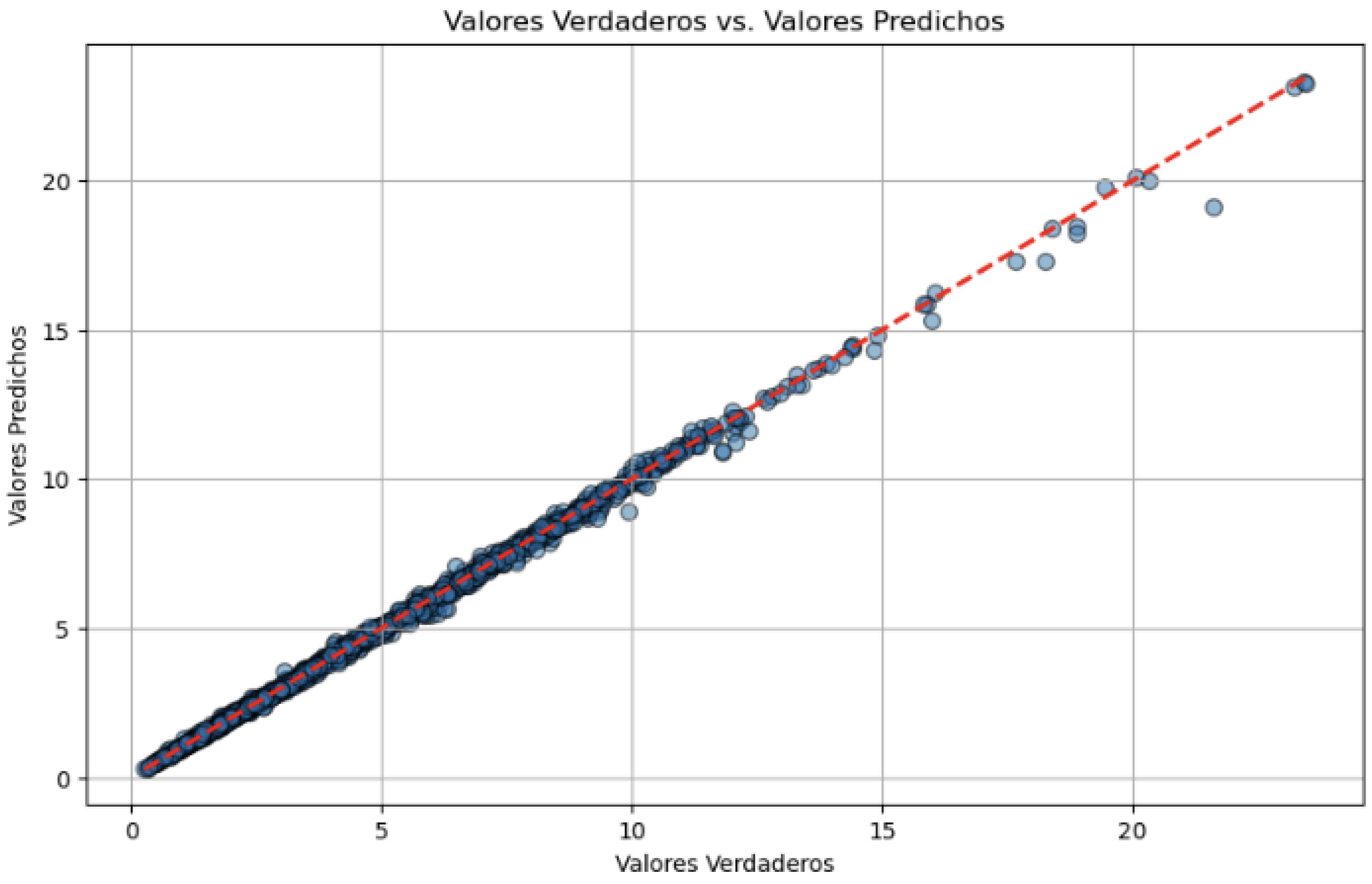

Mapa de Correlación
Gdp_pc_$
Population Thousands
Country Class
Fish & Seafood
Poultry Meat
Pig Meat
Beef Meat
Lamb & Mutton
Other Meat
Eggs
Dairy
Total Protein (capita/day)
Total CO2 (capita/day)
1 -0.015 0.51 0.4 0.63 0.58 0.36 0.15 0.076 0.66 0.64 0.77 0.61
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

```python
# Import different models
from sklearn.linear_model import Ridge, Lasso
from sklearn.tree import DecisionTreeRegressor
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor

# Create pipelines with different models
ridge_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('ridge', Ridge())
])

lasso_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('lasso', Lasso())
])

rf_pipeline = Pipeline([
    ('rf', RandomForestRegressor())
])

# Evaluate Ridge model
ridge_cv_scores = cross_val_score(ridge_pipeline, X, y, cv=5
ridge_mse_scores = -ridge_cv_scores
ridge_mean_mse = ridge_mse_scores.mean()
ridge_mean_rmse = np.sqrt(ridge_mean_mse)
```

```
Ridge Mean MSE: 1.6934788604053307e-07
Ridge Mean RMSE: 0.00041151899839561 85
Ridge Std MSE: 1.3769596257742221e-07
Lasso Mean MSE: 1.448050762449726
Lasso Mean RMSE: 1.2033498088460088
Lasso Std MSE: 0.0
Random Forest Mean MSE: 0.23482873115016506
Random Forest Mean RMSE: 0.48459130321350696
Random Forest Std MSE: 0.1952946922849877
```

```python
from sklearn.model_selection import RandomizedSearchCV

# Define parameter distribution for RandomForestRegressor
param_dist = {
    'rf__max_depth': [None, 10, 20, 30, 40, 50],
    'rf__min_samples_split': [2, 5, 10],
    'rf__min_samples_leaf': [1, 2, 4],
    'rf__max_features': [None, 'sqrt', 'log2']
}

# Create RandomizedSearchCV object
random_search = RandomizedSearchCV(rf_pipeline, param_dist, n_iter=10, cv=5, scoring='neg_mean_squared_error', random_state=42)

# Fit the random search
random_search.fit(X, y)

# Get the best parameters and best score
best_params_random = random_search.best_params_
best_score_random = -random_search.best_score_

# Display the best parameters and best score
print(best_params_random)
print(best_score_random)
```

```
{'rf__min_samples_split': 5, 'rf__min_samples_leaf': 1, 'rf__max_features': None, 'rf__max_depth': 20}
0.24015373058290948
```


Valores Verdaderos vs. Valores Predichos
Valores Predichos
Valores Verdaderos
0
5
10
15
20
0
5
10
15
20

Aprendizajes

- Revisar el tratamiento de los valores nulos
- Mejorar la selección de características
- Probar con más modelos
- Más pruebas con temporalidad

Gracias

Alba Sentís - CodeOp Final Project