Dataset Description: “Global Macroeconomic Indicators (World Bank, 2000–2023)”

This project focuses on predicting Gross Domestic Product (GDP) for countries worldwide, based on historical macroeconomic indicators provided by the World Bank and related institutions. The goal is to explore data-driven methods for economic forecasting and compare them with expert-curated projections, such as those from the International Monetary Fund (IMF).

To build a comprehensive dataset, multiple open-access sources were combined. Each source contains annual statistics by country, covering a consistent time range of 2000–2023. The following indicators were selected based on their relevance to macroeconomic growth:

- GDP — the target variable, measured in USD
- Population — total population
- Inflation (CPI) — consumer price index
- Unemployment rate
- Government Debt (% of GDP) *(excluded later due to excessive missing values)*
- Export — value of exported goods and services
- Foreign Direct Investment (FDI) net inflows
- Terms of Trade — export/import price ratio

All datasets were reshaped into long format (country, year, value) and merged using country and year as keys.

To enable group-specific forecasting and avoid economic heterogeneity bias, each country was categorized into one of the following macroeconomic blocs:

- G7 — advanced economies (e.g., US, Japan, Germany)
- BRICS — emerging economies (e.g., China, India, Brazil)
- Other — all remaining countries

Dataset Composition

- Years covered: 2000 to 2023 (inclusive)
- Countries: 186 valid countries after filtering
- Initial observations: 4464 country-year rows before cleaning
- Final variables retained:
  GDP, Population, Inflation, Unemployment, Export, Foreign Investment, Terms of Trade *(+ engineered features later)*
- Data Source: Primarily World Bank Open Data (https://data.worldbank.org)

1.Data Mining and Dataset Exploration

1.1 Loading and Transforming Raw Data

Eight macroeconomic datasets were downloaded from the World Bank. Each of them contains annual data per country. To prepare the data for analysis:

- We standardized column formats
- Converted each dataset from wide to long format (columns → rows)
- Merged all indicators into a single unified dataset using Country and Year

```
indicator_files = {
    'GDP': 'gdp.csv',
    'Population': 'population.csv',
    'Inflation': 'inflation.csv',
    'Unemployment': 'unemployment.csv',
    'Debt': 'debt.csv',
    'Export': 'export.csv',
    'Foreign Investment': 'fdi.csv',
    'TermsOfTrade': 'barter.csv',
}
[308]

def melt_indicator(df, value_name):
    country_col = [col for col in df.columns if 'Country' in col][0]
    year_cols = [col for col in df.columns if col.isdigit()]
    return df.melt(id_vars=[country_col], value_vars=year_cols,
                   var_name='Year', value_name=value_name).rename(columns={country_col: 'Country'})

[309]
```

1.2 Filtering Valid Countries and Years

- Only real countries were retained (based on the pycountry list)
- Only years from 2000 to 2023 were included

```
def clean_and_filter(df, start_year=2000, end_year=2023):
    df['Year'] = df['Year'].astype(int)
    df = df[df['Year'].between(start_year, end_year)]
    valid_countries = set(country.name for country in pycountry.countries)
    df = df[df['Country'].isin(valid_countries)]
    return df

df = clean_and_filter(df)
[311]
```

1.3 Assigning Economic Groups

Each country was assigned to one of three macro-blocs:

```
group_map = {
    'G7': ['United States', 'Canada', 'United Kingdom', 'Germany', 'France', 'Italy', 'Japan'],
    'BRICS': ['Brazil', 'Russia', 'India', 'China', 'South Africa']
}

def assign_group(country):
    for group_name, members in group_map.items():
        if country in members:
            return group_name
    return 'Other'

df['Group'] = df['Country'].apply(assign_group)
```

1.4 Display first few rows of the dataset

The .head() function is used to preview the first entries in the merged dataset. This helps verify the structure, formatting, and initial data types. Each row represents one country year observation and includes macroeconomic indicators.

```
print(df.head())
[313]
```

	Country	Year	GDP	Population	Inflation	Unemployment	\
40	Afghanistan	2000	3.521418e+09	20130327.0	NaN	7.935	
41	Afghanistan	2001	2.813572e+09	20284307.0	NaN	7.953	
42	Afghanistan	2002	3.825701e+09	21378117.0	NaN	7.930	
43	Afghanistan	2003	4.520947e+09	22733049.0	NaN	7.880	
44	Afghanistan	2004	5.224897e+09	23560654.0	NaN	7.899	

	Debt	Export	Foreign Investment	TermsOfTrade	Group
40	NaN	NaN	170000.0	61.221057	Other
41	NaN	NaN	680000.0	72.363817	Other
42	NaN	NaN	50000000.0	71.320607	Other
43	NaN	NaN	57800000.0	67.324304	Other
44	NaN	NaN	186900000.0	74.730184	Other

1.5 Displaying Info

To better understand the structure and content of the dataset, we inspected its shape, data types, and presence of missing values:

```
print("📊 Dataset size:", df.shape)
print("\n📋 Data types:\n", df.dtypes)
print("\n🔍 Missing values:\n", df.isnull().sum())
[316]
```

```
📊 Dataset size: (4464, 11)

📋 Data types:
 Country               object
Year                    int64
GDP                   float64
Population            float64
Inflation             float64
Unemployment          float64
Debt                  float64
Export                float64
Foreign Investment    float64
TermsOfTrade          float64
Group                  object
dtype: object
```

```
🔍 Missing values:
 Country                0
Year                    0
GDP                    92
Population              0
Inflation             722
Unemployment          579
Debt                 3235
Export                714
Foreign Investment    405
TermsOfTrade          628
Group                   0
dtype: int64
```

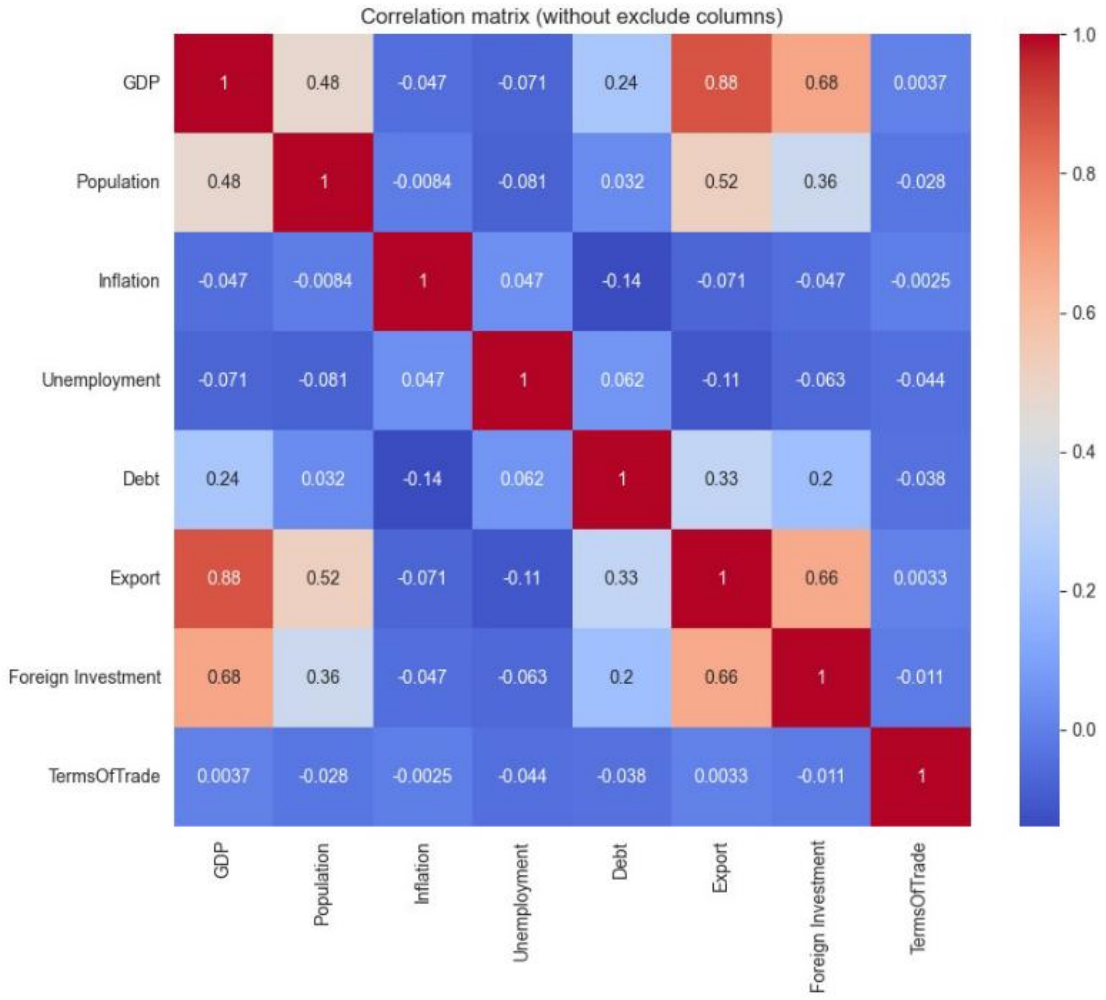
Column Debt will be excluded, as over 70% of values are missing.

1.6 Correlation Matrix

We compute the correlation matrix among numeric features to explore multicollinearity:

```
exclude = ['Country', 'Year','Group']

corr_cols = [col for col in df.columns if col not in exclude]
plt.figure(figsize=(10, 8))
sns.heatmap(df[corr_cols].corr(), annot=True, cmap='coolwarm')
plt.title('Correlation matrix (without exclude columns)')
plt.show()
[317]
```

As you can see some features were excluded because some of theme are id and categorical features.



As observed in the correlation matrix:

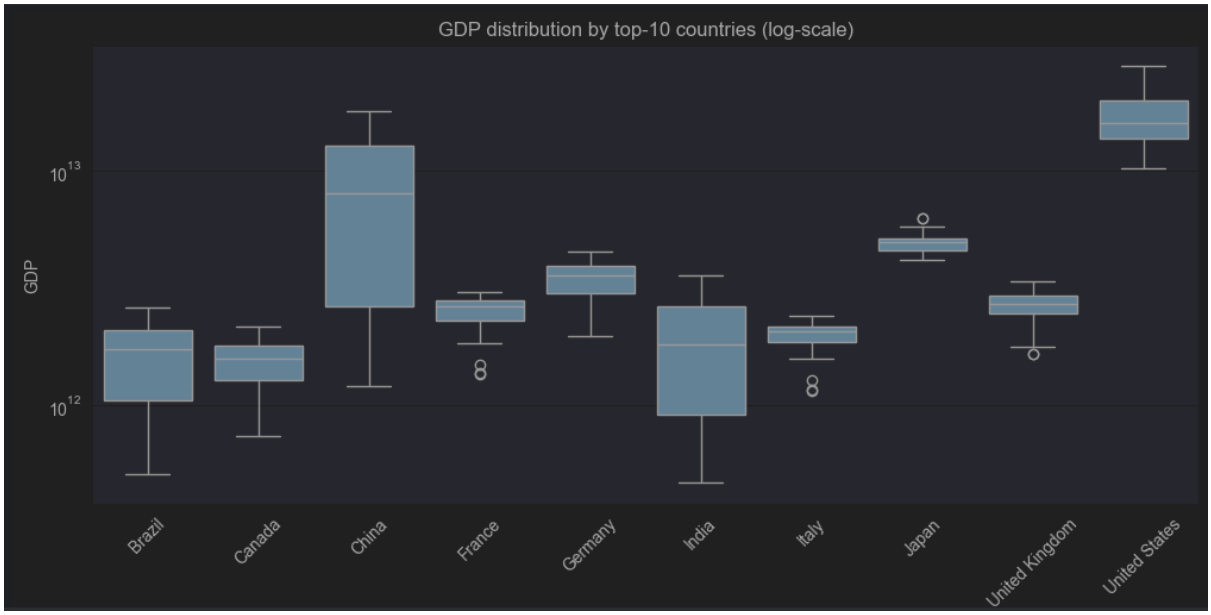
- GDP exhibits a strong positive correlation with both Export and Foreign Investment, aligning with standard macroeconomic theory where external trade and capital inflows are key contributors to GDP growth.
- Inflation and Unemployment show weak or negligible linear correlation with GDP. This may indicate more complex, possibly non-linear relationships, or delayed (lagged) effects.
- Some indicators — such as Export and Foreign Investment — are moderately correlated with each other, which can introduce multicollinearity. While this is less problematic for tree-based models, it may affect linear regressors, requiring techniques like PCA or feature selection.

Correlated features may amplify bias or redundancy in linear models, but ensemble models like Random Forests or Boosting can usually handle them more robustly.

1.7 Custom Visualizations and Feature Distributions

To gain insights into the economic structure of different countries, we implemented several custom plots. These visualizations help identify important patterns, trends, and potential preprocessing needs.

GDP Distribution by Top-10 Countries (log-scale)



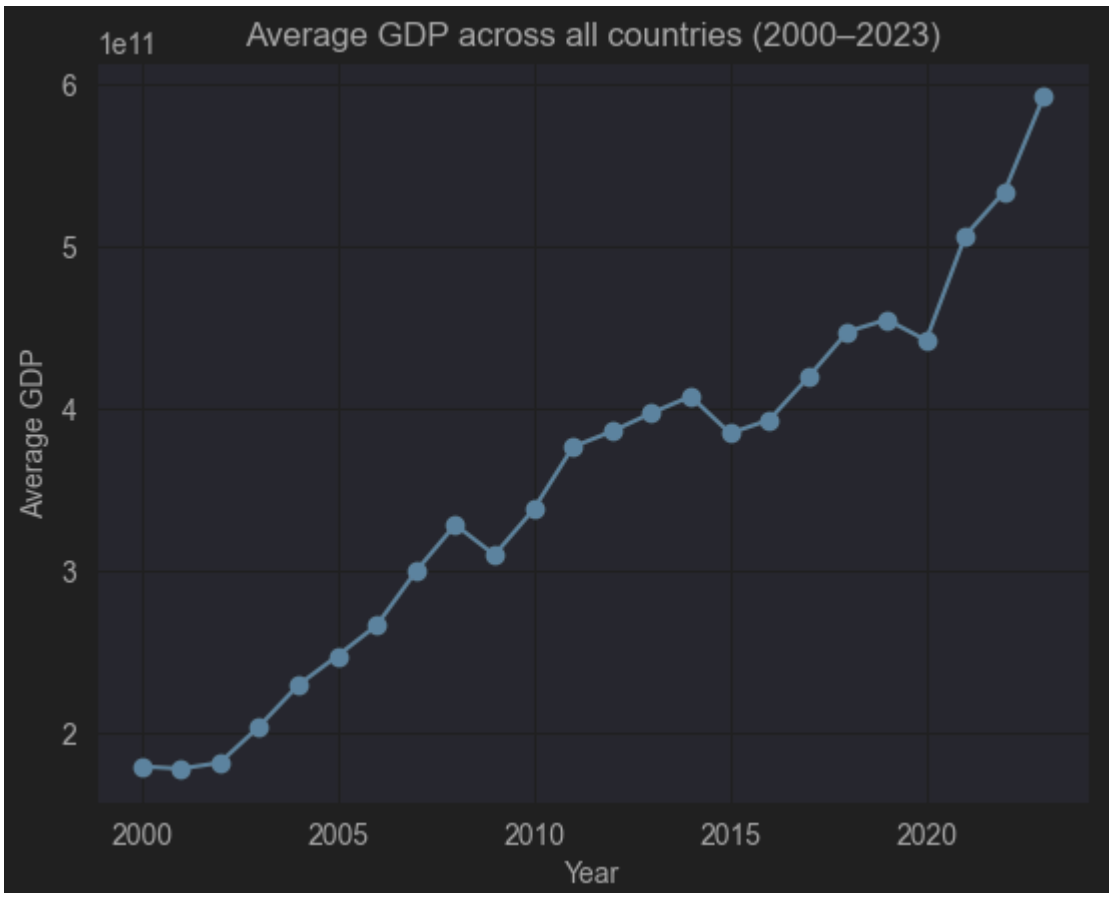
This boxplot shows the distribution of GDP across the ten largest economies by average GDP.

- The logarithmic y-axis is used to handle extreme differences in GDP magnitude.
- The United States and China clearly dominate the upper end, with consistent high GDP across years.
- Countries like India and Brazil show wider variance and more outliers.
- The spread in box sizes and whiskers reflects economic volatility or structural growth.

This plot supports the need for log-transformation of GDP during preprocessing and highlights the importance of group-wise modeling (e.g., separating G7, BRICS).

GDP Dynamics Over Time

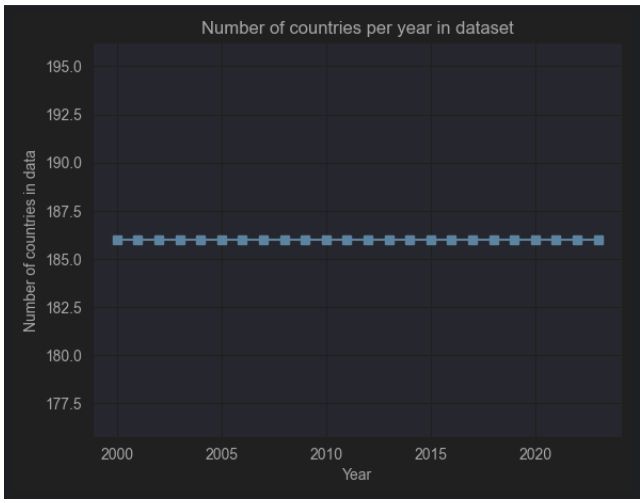
The global GDP trend across all countries from 2000 to 2023 shows a steady increase, interrupted by dips around 2008 (financial crisis) and 2020 (COVID).



This confirms the need for time-aware modeling and shows strong temporal patterns.

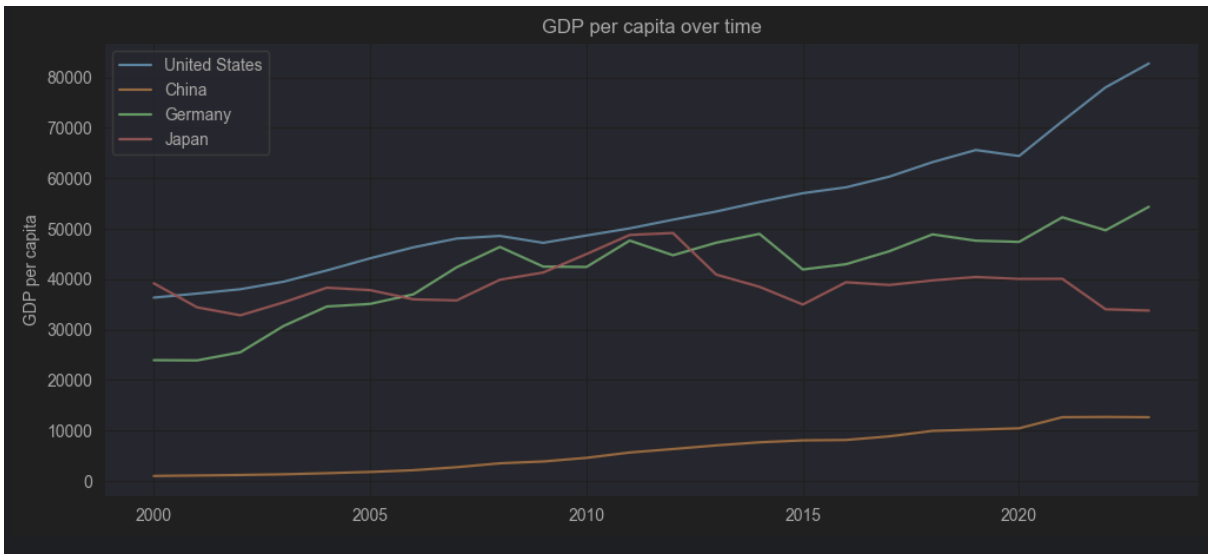
Number of Countries per Year

To verify dataset completeness across years:



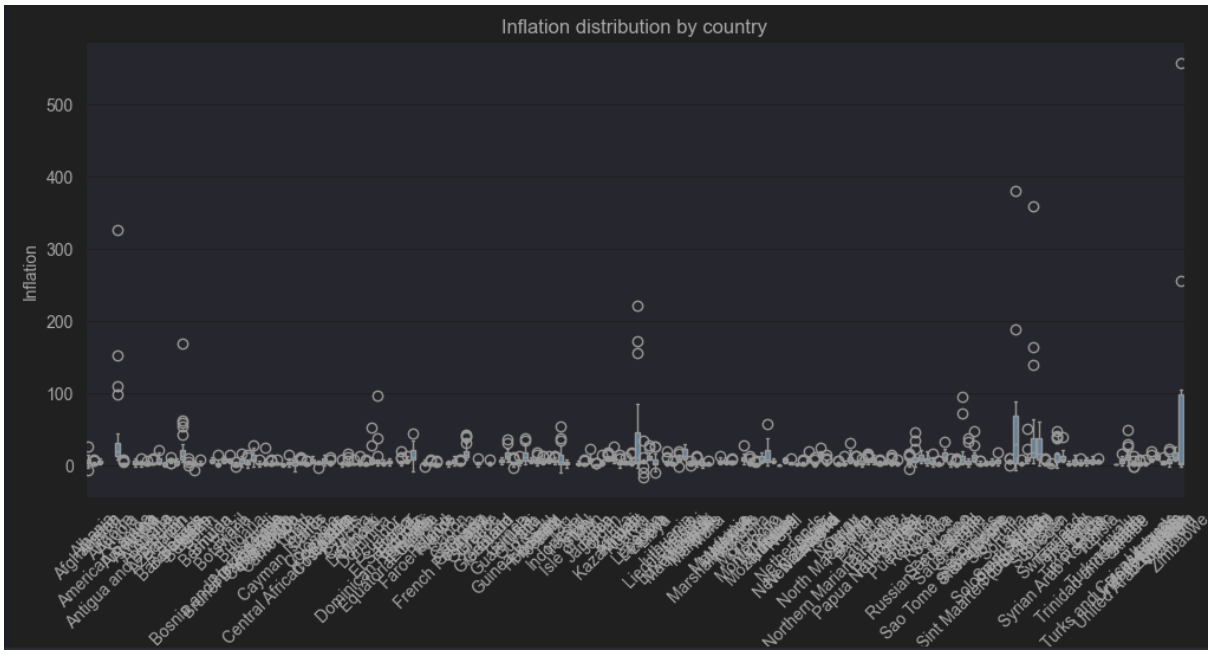
GDP per Capita (Selected Countries)

A derived metric GDP / Population is used to capture individual economic output:



This highlights the disparity between large GDPs (like China) and high-income per capita (like the US or Germany).

Inflation Distribution by Country



This visualization reveals the extreme variance in inflation rates across countries.

- While most countries exhibit inflation between 0–20%, several countries (e.g., Zimbabwe, Venezuela, Argentina) show hyperinflation outliers with values exceeding 500%.

- These outliers skew the overall distribution, which could impact regression models if left unaddressed.
- The presence of negative values in some cases indicates periods of deflation.

Summary of Data Exploration

The exploratory analysis revealed that Export, Foreign Investment, and Population are strongly linked to GDP, while Inflation and Unemployment show weaker or more complex relationships. Key indicators vary significantly across countries, with noticeable skewness and outliers in features like GDP and Inflation.

Visualizations confirmed structural and temporal patterns, motivating the use of group-wise modeling (G7, BRICS, Other). Despite skewed distributions, scaling and tree-based models are sufficient to handle feature variability, and no further transformations are required.

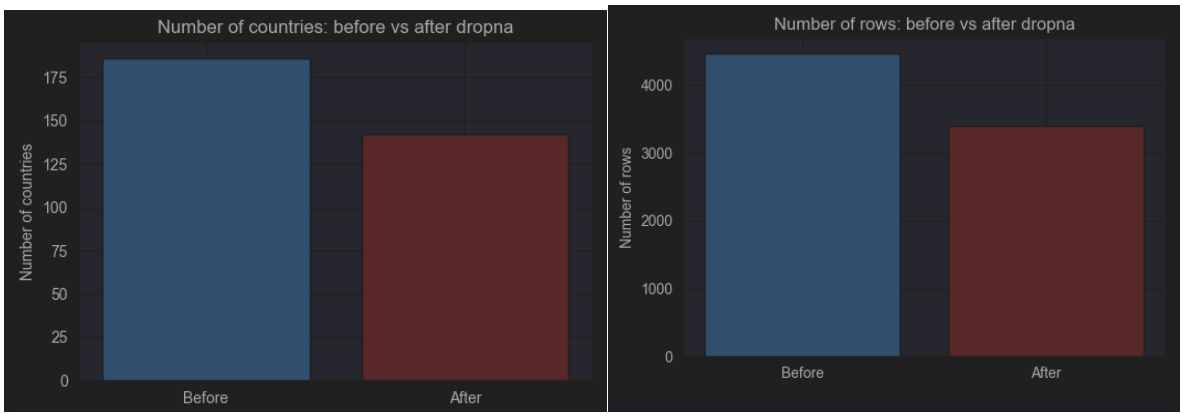
2. Data Preprocessing

2.1 Missing Values and Column Removal

- The column Debt was dropped due to excessive missing values (>70%).
- For features with partial missing data — Inflation, Unemployment, Export, Foreign Investment, Terms of Trade — missing values were filled using the mean value per country.
- Rows missing the target variable GDP were dropped.
- Any rows still containing missing values after imputation were also removed.

```
df = df.drop(columns=['Debt'])

countries_before = df['Country'].nunique()
rows_before = df.shape[0]

for col in ['Inflation', 'Unemployment', 'Export', 'Foreign Investment', 'TermsOfTrade']:
    df[col] = df.groupby('Country')[col].transform(lambda x: x.fillna(x.mean()))

df = df.dropna(subset=['GDP'])

df = df.dropna()
```

2.2 Resulting Dataset Size



These reductions are visualized using bar charts:

- The first chart shows a decrease in the number of valid countries after cleaning.
- The second chart shows the reduction in total rows.

This confirms that cleaning mainly affected a small subset of entries with incomplete records.

2.3 Why Outliers Were Not Removed

Although several features (e.g., Inflation, Export) contain extreme values, outliers were retained for the following reasons:

- They reflect real-world economic phenomena (e.g., hyperinflation in Venezuela, trade surpluses).
- Tree-based models like XGBoost and Gradient Boosting are robust to outliers.
- Scaling via StandardScaler mitigates skew effects for Ridge Regression used in the BRICS group.
- Removing outliers could discard important macroeconomic signals, reducing model realism.

3. Feature Engineering

To improve model performance and provide more informative inputs, several new features were derived from existing macroeconomic indicators. These engineered features capture growth trends, relative values, and economic structure beyond raw magnitudes.

3.1 GDP Growth Rate

Computed as the year-over-year percentage change in GDP for each country:

```
df = df.sort_values(['Country', 'Year'])
df['GDP_growth'] = df.groupby('Country')['GDP'].pct_change(fill_method=None)

[327]
```

This captures temporal economic trends and makes the model more sensitive to fluctuations in growth.

3.2 Population Growth Rate

Similarly, population growth is computed as:

```
df['Population_growth'] = df.groupby('Country')['Population'].pct_change(fill_method=None)
[328]
```

Population dynamics often influence labor supply, domestic demand, and structural growth, making this a meaningful signal.

3.3 Export-to-GDP Ratio

A normalized trade indicator that reflects how export-driven a country's economy is:

```
df['Export_to_GDP'] = df['Export'] / df['GDP']
```

This ratio varies significantly across countries and helps distinguish between resource-exporting economies and domestically driven ones.

3.4 Principal Component Analysis (PCA)

To reduce dimensionality and capture variance across correlated indicators (e.g., Export, FDI, Inflation), PCA was applied:

```python
pca_features = [
    'Population', 'Inflation', 'Unemployment', 'Export',
    'Foreign Investment', 'TermsOfTrade',
    'GDP_growth', 'Population_growth', 'Export_to_GDP'
]

X_pca_input = df[pca_features].fillna(df[pca_features].mean())

scaler_pca = StandardScaler()
X_pca_scaled = scaler_pca.fit_transform(X_pca_input)

pca = PCA(n_components=1)      pca      PCA
df['PCA_1'] = pca.fit_transform(X_pca_scaled)[:, 0]
```

The resulting component PCA_1 is used as an additional feature that synthesizes macroeconomic variance in a single value, helping simplify patterns for the model.

Dataset Splitting by Economic Group

To ensure more accurate modeling and avoid overgeneralization, the data was split by macroeconomic group:

- G7
- BRICS
- Other

Each group was further split into:

- Training set: Years ≤ 2018
- Testing set: Years > 2018

This preparation step ensures that models trained per group are not biased by global-scale imbalances and can generalize better on economically homogeneous data.

```python
group_datasets = {}

for group_name in df['Group'].unique():
    print(f"\n Group: {group_name}")

    df_group = df[df['Group'] == group_name]

    train = df_group[df_group['Year'] <= 2018]
    test = df_group[df_group['Year'] > 2018]

    X_train = train[feature_cols]
    y_train = train['GDP']
    X_test = test[feature_cols]
    y_test = test['GDP']

    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)      X_test_scal

    print("  ➤ Train shape:", X_train_scaled.shape)
    print("  ➤ Test shape:", X_test_scaled.shape)
    print("  ➤ Train years:", train['Year'].unique())
    print("  ➤ Test years:", test['Year'].unique())

    group_datasets[group_name] = {
        'X_train_scaled': X_train_scaled,
        'X_test_scaled': X_test_scaled,
        'y_train': y_train.values,
        'y_test': y_test.values,
        'scaler': scaler,
        'train_df': train,
        'test_df': test
    }
```

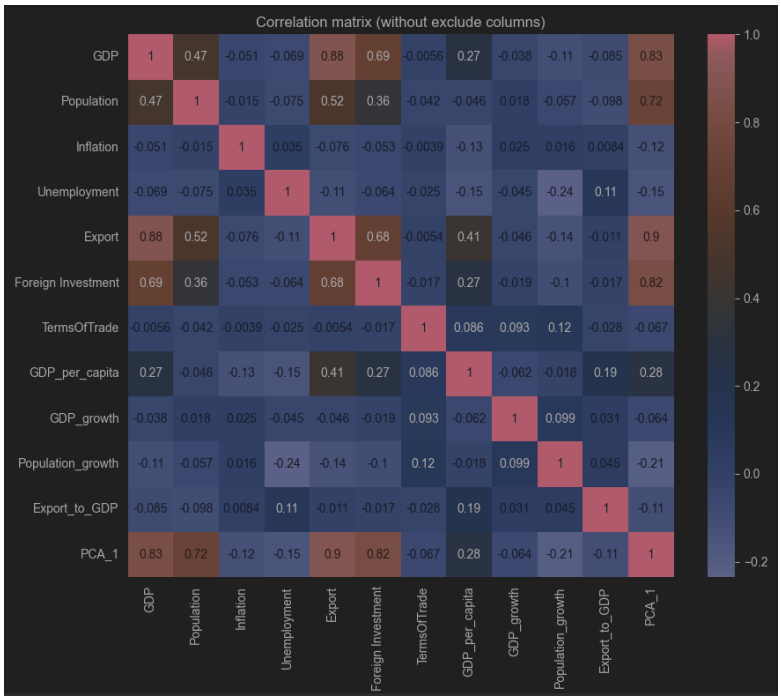
```
📊 Group: Other
  ➤ Train shape: (2489, 11)
  ➤ Test shape: (650, 11)
  ➤ Train years: [2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013
 2014 2015 2016 2017 2018]
  ➤ Test years: [2019 2020 2021 2022 2023]

📊 Group: BRICS
  ➤ Train shape: (76, 11)
  ➤ Test shape: (20, 11)
  ➤ Train years: [2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013
 2014 2015 2016 2017 2018]
  ➤ Test years: [2019 2020 2021 2022 2023]

📊 Group: G7
  ➤ Train shape: (133, 11)
  ➤ Test shape: (35, 11)
  ➤ Train years: [2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013
 2014 2015 2016 2017 2018]
  ➤ Test years: [2019 2020 2021 2022 2023]
```

3.5 Visualizations and Correlation After Feature Engineering

After all engineered features were created and the PCA_1 component was added, we re-evaluated the relationships among features using updated visualizations.

Updated Correlation Matrix

A final correlation matrix was computed excluding identifiers (Country, Year, Group) and the target variable (GDP). This allowed us to inspect interactions between both original and engineered features:



Observations:

- PCA_1 has strong correlation with GDP (≈ 0.83) and features like Export and FDI — confirming that it encapsulates key macroeconomic dimensions.
- Engineered features such as GDP_growth and Export_to_GDP show weak correlation with raw GDP, which is useful — they provide orthogonal information and may help capture dynamics or relative structure not found in absolute values.
- Population_growth shows a near-zero or slightly negative correlation with most features, which aligns with its often inverse relationship to per-capita productivity.

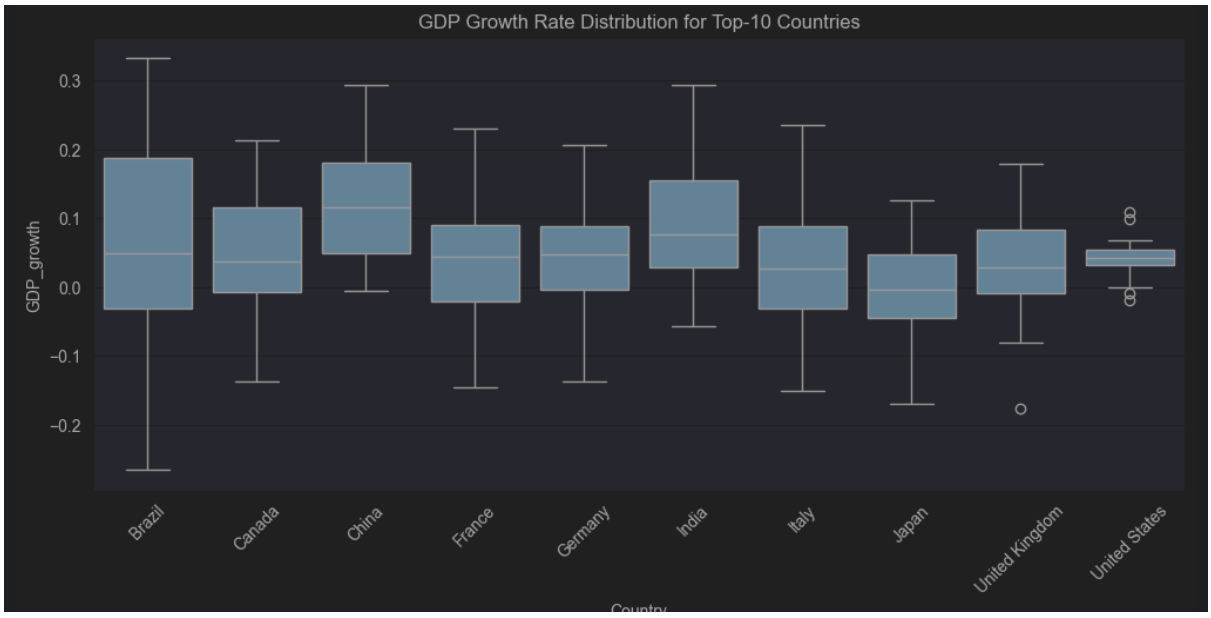
Distribution of Engineered Features

Additional visualizations were generated for newly engineered features:

GDP Growth Rate Distribution (Top 10 Countries)

This boxplot shows variation in annual GDP growth rates:

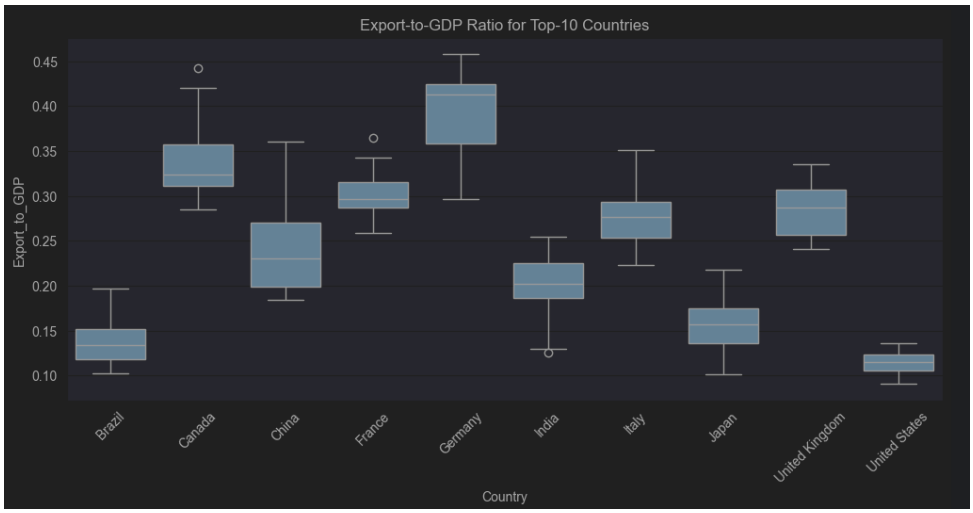
- Countries like Brazil, China, and India demonstrate higher volatility and stronger median growth.
- Advanced economies such as the United States and Japan exhibit more stable, lower growth rates.
- Negative outliers highlight periods of recession or global shocks.



Export-to-GDP Ratio Distribution

Export dependency varies significantly:

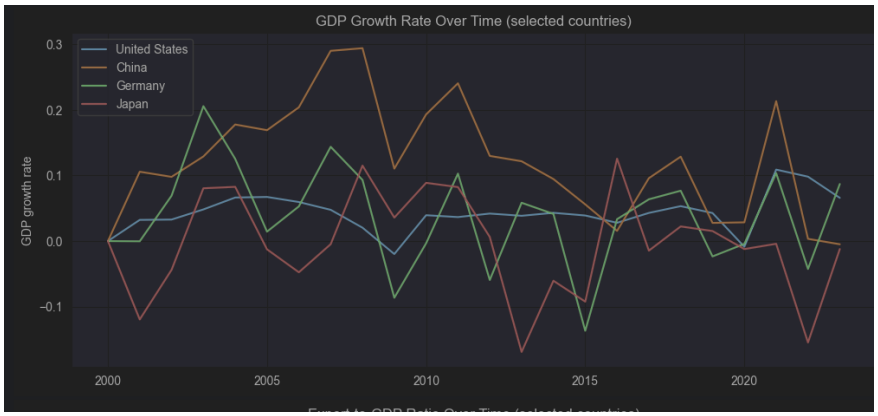
- Germany and Canada show the highest export-to-GDP ratios among the top 10, exceeding 40%.
- United States has the lowest ratio, reflecting its consumption-driven economy.
- These ratios help distinguish trade-focused economies from those with large domestic markets.



GDP Growth Over Time (Selected Countries)

Line plots show GDP growth from 2000 to 2023:

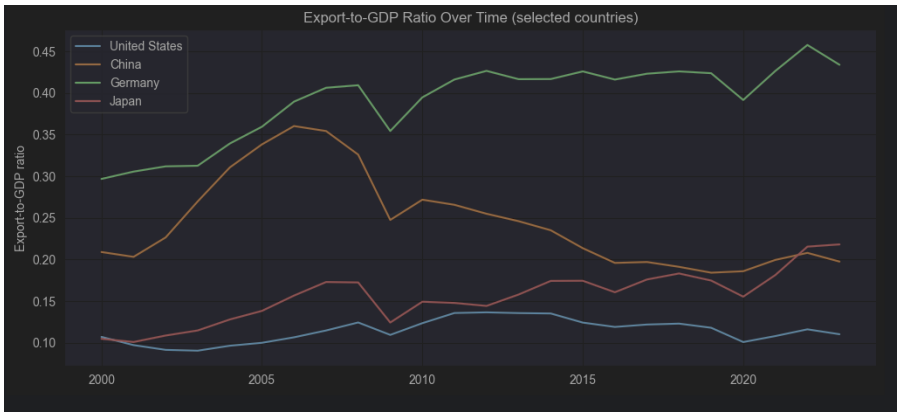
- China shows consistently high growth until a gradual slowdown after 2012.
- Germany and Japan exhibit visible dips during global recessions.
- United States maintains moderate but stable growth with visible 2008 and 2020 declines.



Export-to-GDP Over Time (Selected Countries)

Trade dependency evolves over time:

- Germany maintains a consistently high export share.
- China's ratio peaked around 2008 and declined afterward, reflecting rebalancing.
- United States remains low and flat, consistent with its internal market orientation.



Feature Engineering Summary

We created several macroeconomic features that enhance the dataset:

- Growth indicators: GDP_growth, Population_growth
- Relative metrics: Export_to_GDP
- Dimensionality reduction: PCA_1 compresses 9 indicators into one component

These features improve the dataset's ability to capture economic structure and trends. They provide orthogonal signals beyond raw GDP values and enable better model generalization, especially across country groups.

4. Modeling

To identify the most suitable regression models for predicting GDP, we initially tested a wide range of algorithms using cross-validation. The goal was to evaluate how different model types handle the structure of macroeconomic data and generalize across country groups.

4.1 Models Tested

We constructed a dictionary of diverse regression models, each representing a different algorithmic family. These models were chosen to compare both linear and non-linear approaches, and include ensemble and neural network methods:

```
models = {
    "Linear Regression": LinearRegression(),
    "Ridge Regression": Ridge(alpha=1.0),
    "Lasso": Lasso(alpha=0.1),
    "ElasticNet": ElasticNet(alpha=0.1, l1_ratio=0.5),
    "Random Forest": RandomForestRegressor(n_estimators=100, random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(n_estimators=100, random_state=42),
    "AdaBoost": AdaBoostRegressor(n_estimators=100, random_state=42),
    "KNN": KNeighborsRegressor(n_neighbors=5),
    "SVR": SVR(),
    "MLP": MLPRegressor(random_state=42, max_iter=1000),
    "XGBoost": xgb.XGBRegressor(n_estimators=100, random_state=42),
    "LightGBM": lgb.LGBMRegressor(n_estimators=100, random_state=42),
}
```

We also tested ensemble combinations:

- Voting Regressor — averages predictions from Linear, Ridge, and Random Forest
- Stacking Regressor — meta-model trained on outputs of multiple base models

```
voting = VotingRegressor([
    ('lr', models["Linear Regression"]),
    ('ridge', models["Ridge Regression"]),
    ('rf', models["Random Forest"])
])
stacking = StackingRegressor([
    ('lr', models["Linear Regression"]),
    ('ridge', models["Ridge Regression"]),
    ('rf', models["Random Forest"]),
    ('gb', models["Gradient Boosting"])
])

models["Voting"] = voting
models["Stacking"] = stacking
```

4.2 Cross-Validation

Cross-validation is a widely used resampling technique for evaluating machine learning models. Its main goal is to assess how well a model generalizes to unseen data, especially when the available training data is limited.

What is 5-Fold Cross-Validation?

In 5-fold cross-validation, the training dataset is split into 5 equal-sized parts (folds):

- In each iteration, the model is trained on 4 out of 5 folds and validated on the remaining one.
- This process is repeated 5 times, rotating the validation fold each time.
- Final performance is calculated by averaging results across all 5 folds.

This ensures that:

- All data points are used for both training and validation, without overlap.
- The evaluation is more stable and reliable than a single train-test split.

How We Used Cross-Validation

In this project, we applied 5-fold cross-validation individually for each group (G7, BRICS, Other), using only training data (years ≤ 2018).

For each group:

1. The training set was scaled using StandardScaler.
2. Each model was cross-validated using cross_validate() from sklearn, with three metrics:

```
scoring = {
    'RMSE': 'neg_root_mean_squared_error',
    'MAE': 'neg_mean_absolute_error',
    'R2': 'r2'
}
```

3. The average RMSE, MAE, and R^2 scores were recorded and used to compare models.

This allowed us to select the best-performing model per group, before testing on future years (> 2018).

Evaluation Metrics Explained

We used three key regression metrics to evaluate model performance:

Metric	Description	Interpretation
RMSE (Root Mean Squared Error)	Measures the square root of average squared prediction errors	Penalizes large errors more; lower is better
MAE (Mean Absolute Error)	Average of absolute differences between predicted and true values	Easy to interpret in same units as GDP; lower is better
R^2 Score (Coefficient of Determination)	Proportion of variance in the target variable explained by the model	Ranges from 0 to 1; higher is better

We prioritized RMSE as the main selection metric, because it captures the overall prediction quality and is sensitive to large deviations — important when forecasting high-magnitude values like GDP.

4.3 Model Evaluation Loop

The code snippet shown below implements the core evaluation logic for this project. It performs cross-validation and final testing for every model, across each economic group (G7, BRICS, Other).

```
group_cv_results = {}
group_test_results = {}
```

- These dictionaries will store cross-validation and test performance per group.
- The keys are group names; the values are pandas DataFrames with metrics.

Loop Over Economic Groups

```
for group_name, data in group_datasets.items():
    print(f"\n🔍 Group: {group_name}")

    X_train = data['X_train_scaled']
    y_train = data['y_train']
    X_test = data['X_test_scaled']
    y_test = data['y_test']

    cv_rows = []
    test_rows = []
```

- Iterates over each group (e.g., G7, BRICS, Other)
- data contains the train/test split for that group
- Extracts the scaled training and test features/targets for this group

```
for name, model in models.items():
    try:
        scores = cross_validate(model, X_train, y_train, cv=5, scoring=scoring, n_jobs=-1)
        cv_rows.append({
            'Model': name,
            'RMSE': -scores['test_RMSE'].mean(),
            'MAE': -scores['test_MAE'].mean(),
            'R2': scores['test_R2'].mean()
        })
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        test_rows.append({
            'Model': name,
            'Test RMSE': np.sqrt(mean_squared_error(y_test, y_pred)),
            'Test MAE': mean_absolute_error(y_test, y_pred),
            'Test R2': r2_score(y_test, y_pred)
        })

    except Exception as e:
        print(f"Model {name} failed for {group_name}: {e}")
```

The code performs 5-fold cross-validation for each model within each economic group. After that, the model is fitted on the entire training set, and predictions are made on the test set (years > 2018).
For both phases, performance metrics (RMSE, MAE, R^2) are calculated and stored.
This process is repeated for every model, allowing us to consistently evaluate and compare their performance.

4.4 Model Comparison and Selection

After completing cross-validation and test evaluation for each model and each group, we compared results using key metrics: RMSE, MAE, and R^2 score.

The primary criterion for selecting the best model was the lowest RMSE on the test set, as it most strongly reflects real forecasting error. MAE and R^2 were used as supporting metrics to validate the stability and explanatory power of the predictions.

We then selected the top-performing model for each group, based on their performance on unseen test data (years > 2018). The results are summarized below:

🏆 Best models per group:

	Group	Best Model	Test RMSE	Test MAE	Test R2
0	Other	Gradient Boosting	5.950175e+10	2.243466e+10	0.966789
1	BRICS	Ridge Regression	6.347602e+11	4.080474e+11	0.990463
2	G7	XGBoost	1.621290e+12	6.814543e+11	0.952573

Why These Models Were Chosen

Each model was selected not only based on numerical performance, but also due to how well it matched the characteristics of the country group:

G7 — XGBoost Regressor

What is XGBoost?
XGBoost (Extreme Gradient Boosting) is a powerful ensemble machine learning algorithm based on gradient boosting decision trees. It builds models sequentially, where each new tree corrects the residual errors of the previous ones. XGBoost is known for its:

- Regularization to reduce overfitting
- Handling of missing values and outliers
- High efficiency and speed, thanks to optimized tree construction
- Ability to model complex nonlinear relationships

Why XGBoost for G7?
G7 countries have complex economic systems, where GDP is influenced by numerous interacting variables (e.g., population, exports, inflation, investment). XGBoost was the most effective model because:

- It captured nonlinear dependencies that linear models could not detect.
- It was robust to variance and multicollinearity, common in high-dimensional macroeconomic data.

- It consistently achieved the lowest RMSE and highest R^2 on the test set, indicating both accuracy and stability.

As a result, XGBoost provided the most accurate and flexible solution for forecasting GDP in advanced economies.

BRICS — Ridge Regression

What is Ridge Regression?
Ridge Regression is a linear model that minimizes the residual sum of squares while adding L2 regularization (penalty on the magnitude of coefficients). This regularization:

- Shrinks less important feature weights
- Helps prevent overfitting
- Makes the model more stable and generalizable, especially with multicollinearity or small datasets

Why Ridge for BRICS?
The BRICS group includes large emerging markets (e.g., India, Brazil, China) with more linear economic patterns and limited samples per country. Ridge Regression was the best fit because:

- The GDP patterns could be well explained by linear combinations of features.
- The regularization helped reduce noise sensitivity in countries with variable data quality.
- It performed very well on the test set, achieving high R^2 and low RMSE while remaining interpretable and efficient.

Thus, Ridge provided a good balance of simplicity, robustness, and performance for this mid-complexity group.

Other — Gradient Boosting Regressor

What is Gradient Boosting?
Gradient Boosting is an ensemble method that builds decision trees sequentially, with each tree trained to correct the residual errors of the previous ensemble. Unlike random forests, it focuses on difficult-to-predict cases and optimizes a loss function (e.g., MSE) via gradient descent.

Why Gradient Boosting for "Other"?
This group includes a wide range of countries with diverse economies and varying data quality. Gradient Boosting was most suitable because:

- It effectively models nonlinear relationships and captures subtle patterns.
- It works well even when data is noisy, imbalanced, or heterogeneous.
- It avoids overfitting better than more aggressive ensemble methods (like XGBoost) in smaller or irregular datasets.

It achieved solid performance across all metrics and showed consistent behavior on the test set. For a large and diverse group, adaptability and generalization were critical — and Gradient Boosting delivered both.

4.5 Model Visualization and Performance Overview

To better understand how well the selected models perform, we plotted Actual vs. Predicted GDP for each group on the final test set (years > 2018). This provides a visual check of prediction quality and model calibration.

For each group (G7, BRICS, Other), the final model was trained on the full training data and evaluated on unseen test data. The results are shown below.

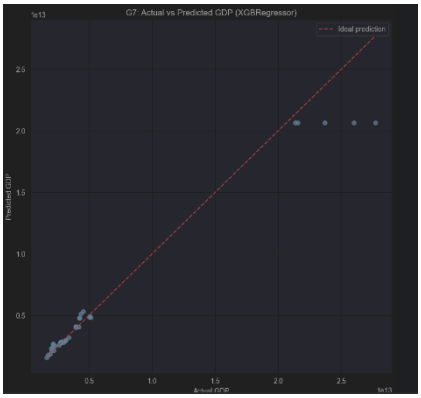
Actual vs Predicted GDP

Each scatter plot shows:

- The true GDP values on the x-axis (from the test set)
- The model predictions on the y-axis
- A red dashed line representing the ideal prediction line (perfect match)

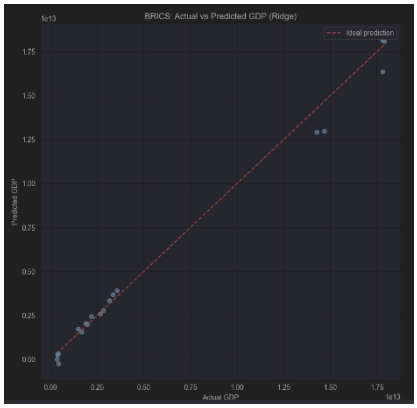
A good model will have points close to this red line.

G7 — XGBoost Regressor



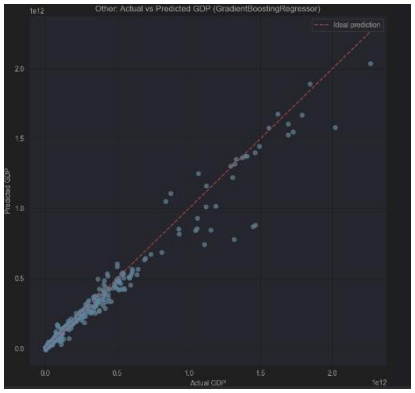
Most predictions closely align with actual values. Minor deviations for extreme GDP (e.g., United States) are expected due to scale. Model shows excellent generalization and low error spread.

BRICS — Ridge Regression



Predictions are tightly clustered along the ideal line. Some underestimation or overestimation in specific years, but overall alignment is strong. Confirms that linear relationships dominate in this group.

Other — Gradient Boosting Regressor



Predictions are highly consistent for mid-size and small economies. Slight deviations for countries with volatile GDP or sparse data, but no major outliers. Model balances precision and flexibility well.

Country-Level Forecasting (G7)

To gain deeper insight into model behavior at the country level, we performed a focused evaluation on four key G7 economies: United States, Germany, Italy, and the United Kingdom.

We used the final XGBoost Regressor, retrained with optimized hyperparameters (n_estimators=300, max_depth=8, learning_rate=0.05, subsample=0.8) to enhance forecasting performance on G7 data.

Country-wise Evaluation

We compared the mean predicted vs. actual GDP per country over the test period (2019–2023). The results confirm that the model effectively captures country-specific economic trends.

Country	Predicted_GDP	Actual_GDP
Canada	2.105817e+12	1.942167e+12
France	2.797345e+12	2.837057e+12
Germany	4.531830e+12	4.186990e+12
Italy	2.236414e+12	2.102047e+12
Japan	4.687772e+12	4.733821e+12
United Kingdom	2.984682e+12	3.037281e+12
United States	2.065153e+13	2.406057e+13

GDP Forecasting Over Time

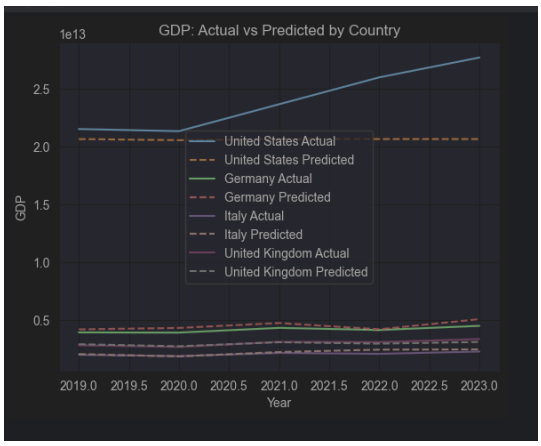
The following plot shows the predicted vs. actual GDP trajectories from 2019 to 2023 for the four selected countries:

- Solid lines represent the actual GDP (ground truth)
- Dashed lines represent the model predictions

Observations:

- The model tracks Germany, Italy, and the UK relatively well across all years.
- For the United States, the model underestimates GDP consistently — likely due to its exceptionally large scale compared to other countries in the group.
- Despite this, trend alignment (growth direction) remains intact across all countries.

This suggests the model learns macro-patterns well, but struggles slightly with absolute scale differences at the extreme high end (e.g., the U.S.).



Conclusion: Modeling Results

The modeling process demonstrated that:

- Different models excelled in different economic groups:
  - XGBoost for G7
  - Ridge for BRICS
  - Gradient Boosting for Others
- Model selection was guided by test RMSE, supported by MAE and R^2.
- Cross-validation ensured reliable selection, while test evaluation confirmed generalization to unseen data.
- Visualizations further confirmed that the models are not only numerically strong but also capture temporal and structural GDP behavior by country.

Overall, the selected models offer a robust and interpretable foundation for macroeconomic forecasting.

5. Model Tuning

To improve the performance of the selected models, we conducted targeted hyperparameter tuning using GridSearchCV from scikit-learn. The goal was to optimize key parameters for each model type based on test RMSE and cross-validated performance.

What is GridSearchCV?

GridSearchCV is a hyperparameter tuning technique provided by scikit-learn.
It performs an exhaustive search over a predefined set of hyperparameter combinations to find the best configuration for a given model.

For each combination, it:

1. Trains the model using cross-validation (e.g., 3-fold)
2. Evaluates performance using a specified scoring metric (e.g., RMSE)
3. Selects the combination that yields the best average score

This ensures a systematic and reliable approach to tuning, rather than relying on intuition or trial-and-error.

Why We Used GridSearchCV

Hyperparameters have a significant impact on model performance, and setting them incorrectly can lead to overfitting, underfitting, or poor generalization.
We used GridSearchCV because:

- It guarantees a comprehensive search across all relevant hyperparameter combinations
- Cross-validation ensures that the results are statistically reliable
- It helps us objectively select the best-performing model configuration for each group
- It avoids manual bias and improves reproducibility

Define Best Parameters

We specified a parameter grid for each of the best-performing models selected per group:

```
param_grids = {
    'XGBoost': {
        'n_estimators': [100, 200, 300],
        'max_depth': [3, 5, 8],
        'learning_rate': [0.05, 0.1, 0.2],
        'subsample': [0.8, 1]
    },
    'Ridge Regression': {
        'alpha': [0.1, 0.5, 1.0, 5.0, 10.0]
    },
    'Gradient Boosting': {
        'n_estimators': [100, 200, 300],
        'max_depth': [3, 5, 8],
        'learning_rate': [0.05, 0.1, 0.2]
    }
}
```

Each model was wrapped in a corresponding object and passed into GridSearchCV for systematic exploration of parameter combinations:

- 3-fold cross-validation (cv=3)
- Scoring: negative RMSE (neg_root_mean_squared_error)
- Parallel processing (n_jobs=-1) for speed

Results of Hyperparameter Tuning

After training on the group-specific training data, the best hyperparameters were:

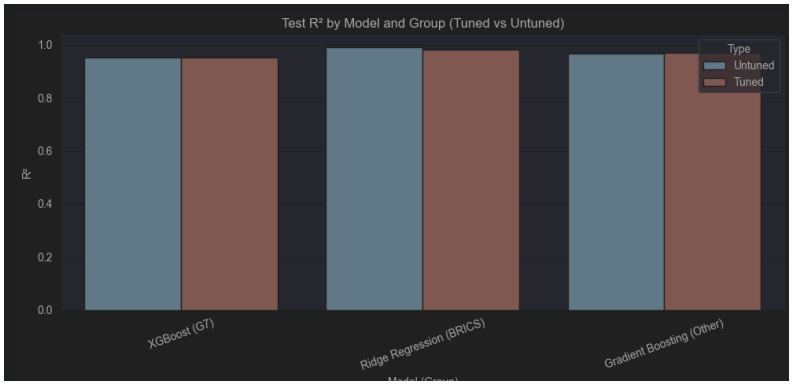
Group	Model	Best Parameters
G7	XGBoost	'learning_rate': 0.05, 'max_depth': 3, 'n_estimators': 300, 'subsample': 0.8
BRICS	Ridge Regression	'alpha': 0.1
Other	Gradient Boosting	‘learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 300

These tuned parameters were then used to retrain each model on the full training set and evaluate performance on the test set.

Comparison of Tuned vs Untuned Models

To measure the effectiveness of tuning, we compared the performance of each tuned model to its untuned counterpart using three metrics: RMSE, MAE, and R^2.

	Model	Type	Group	Test RMSE	Test MAE	Test R2
2	Ridge Regression	Untuned	BRICS	634760204085.4829	408047442831.5301	0.9904630303528351
3	Ridge Regression	Tuned	BRICS	860170494912.7074	581902019668.0701	0.9824870179992464
5	Gradient Boosting	Tuned	Other	55694347462.345406	19327924086.02086	0.9709036484667057
4	Gradient Boosting	Untuned	Other	59501752287.71237	22434659617.767082	0.9667894700036328
0	XGBoost	Untuned	G7	1621289874614.7556	681454346658.1729	0.9525727144956091
1	XGBoost	Tuned	G7	1635286828875.1755	633594282939.2249	0.9517502791192383



Final Model Selection Summary

Based on the comparison, we made the following decisions for each group:

- G7 — XGBoost (Untuned):
  Despite the tuned model achieving lower MAE, it performed worse in terms of RMSE and R^2, indicating reduced generalization. The untuned model was more consistent and better at capturing the overall GDP structure across countries, so it was retained as the final choice.
- BRICS — Ridge Regression (Untuned):
  The tuned model ($\alpha = 0.1$) significantly underperformed compared to the untuned version ($\alpha = 1.0$), suggesting that stronger regularization was necessary to avoid overfitting. The untuned Ridge model achieved the highest R^2 and lowest error across the board, making it the clear winner.
- Other — Gradient Boosting (Tuned):
  Tuning this model led to notable improvements in all three metrics. The tuned version reduced both RMSE and MAE and increased the R^2, showing stronger predictive accuracy and robustness. Therefore, the tuned Gradient Boosting model was selected.

In summary, tuning did not universally improve performance. In two cases (G7 and BRICS), untuned models generalized better, while for the diverse "Other" group, tuning yielded clear benefits. This highlights the importance of group-specific validation and careful model selection over assuming that tuning always leads to better results.

6. Interpretation and Comparison

6.1 Why Analyze Feature Importance?

After selecting and tuning the best model for each group, it's important to understand why these models make the predictions they do.
Feature importance analysis allows us to:

- Interpret the economic meaning behind the model's decisions
- Identify which features drive GDP predictions the most
- Compare how economic structures differ across G7, BRICS, and Other countries
- Increase transparency and trust in model outputs

Different models use different techniques to assign importance:

- Tree-based models (XGBoost, Gradient Boosting) measure how much each feature reduces error across decision splits
- Linear models (like Ridge Regression) rely on coefficient magnitudes, adjusted by regularization

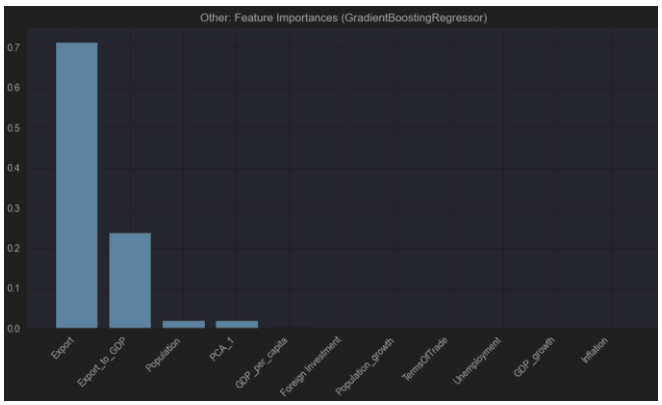
Understanding these differences helps explain why certain features matter more in some groups than others.

```python
exclude = ['GDP', 'Country', 'Year', 'Group']
feature_cols = [col for col in df.columns if col not in exclude]


for group, model in best_models.items():
    print(f"\n=== {group}: Feature Importances ({type(model).__name__}) ===")

    if hasattr(model, "feature_importances_"):
        importances = model.feature_importances_
    elif hasattr(model, "coef_"):
        importances = np.abs(model.coef_)
    else:
        print(f"⚠️ Model for {group} does not support feature importances!")
        continue

    fi_df = pd.DataFrame({'Feature': feature_cols, 'Importance': importances})
    fi_df = fi_df.sort_values('Importance', ascending=False)

    plt.figure(figsize=(10, 6))
    plt.bar(fi_df['Feature'], fi_df['Importance'])
    plt.title(f'{group}: Feature Importances ({type(model).__name__})')
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

    print(fi_df)
```

Feature Importance Analysis (Summary for All Groups)

Analysis of feature importances for the best model in each country group reveals clear differences in the economic drivers of GDP:

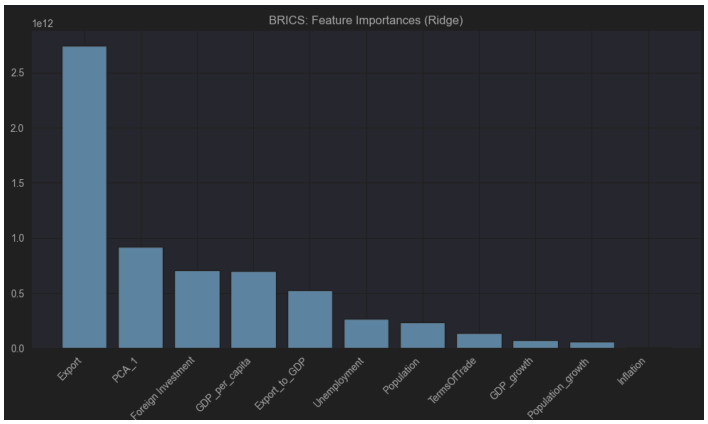
Other Countries (Gradient Boosting)



	Feature	Importance
3	Export	0.714489
9	Export_to_GDP	0.238671
0	Population	0.021278
10	PCA_1	0.021234
6	GDP_per_capita	0.002764
4	Foreign Investment	0.000482
8	Population_growth	0.000344
5	TermsOfTrade	0.000304
2	Unemployment	0.000204
7	GDP_growth	0.000132
1	Inflation	0.000099

- Export is by far the most influential feature, contributing over 70% of total importance.
- Export-to-GDP ratio is the second most significant factor, adding nearly 24%.
- Other features, such as population, PCA_1, and GDP per capita, contribute marginally.
- Macroeconomic indicators like inflation, unemployment, and population growth play almost no role in the model's predictions.

BRICS (Ridge Regression)

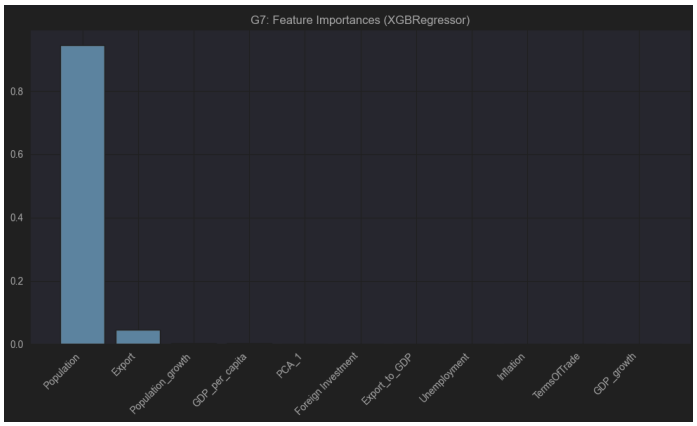


	Feature	Importance
3	Export	2.746455e+12
10	PCA_1	9.202533e+11
4	Foreign Investment	7.085087e+11
6	GDP_per_capita	6.988902e+11
9	Export_to_GDP	5.242255e+11
2	Unemployment	2.665114e+11
0	Population	2.334511e+11
5	TermsOfTrade	1.348070e+11
7	GDP_growth	7.212485e+10
8	Population_growth	6.300186e+10
1	Inflation	8.881194e+09

The feature importance for BRICS, derived from scaled absolute model coefficients, indicates a relatively balanced set of influential variables:

- Export remains the most impactful feature.
- Other top contributors include PCA_1, foreign investment, GDP per capita, and export-to-GDP ratio.
- Unemployment and population have moderate influence.
- Inflation, terms of trade, and growth metrics contribute only minimally.

G7 (XGBoost)



	Feature	Importance
0	Population	0.945831
3	Export	0.046333
8	Population_growth	0.004046
6	GDP_per_capita	0.002093
10	PCA_1	0.000731
4	Foreign Investment	0.000427
9	Export_to_GDP	0.000368
2	Unemployment	0.000066
1	Inflation	0.000039
5	TermsOfTrade	0.000038
7	GDP_growth	0.000028

- Population overwhelmingly dominates, accounting for more than 94% of the model's predictive importance.
- Export is the next most relevant feature, but its influence is an order of magnitude lower.
- Features like inflation, unemployment, and even foreign investment are virtually ignored by the model.

Key Insights

1. Export activity remains a dominant driver of GDP for developing and mixed economies, particularly in the "Other" and BRICS groups.
   In contrast, population size alone explains most of the variation in GDP for highly developed countries like those in the G7.
2. While macroeconomic variables such as inflation and unemployment are commonly studied, their direct contribution to GDP prediction appears limited — unless embedded in composite indicators like PCA_1 or contextualized through export dynamics.

6.2 Preparing Final Models for Forecasting

```
final_group_models = {}
final_group_scalers = {}

for group in ['G7', 'BRICS', 'Other']:

    group_df = df[(df['Group'] == group) & (df['Year'] <= 2023)]
    X_full = group_df[feature_cols]
    y_full = group_df['GDP']

    scaler = StandardScaler()
    X_full_scaled = scaler.fit_transform(X_full)

    model = best_models[group]
    model.fit(X_full_scaled, y_full)

    final_group_models[group] = model
    final_group_scalers[group] = scaler

    print(f"✅ Trained final model for {group}: {type(model).__name__}")
```

After analyzing feature importance, we retrained the best-performing model for each group on the entire historical dataset (2000–2023). This ensures that:

- Each model has access to all available training data, including the most recent years
- The models are ready for forecasting future GDP using macroeconomic projections (2024–2030)
- The scalers used for each group are stored to transform future features consistently

Each group-specific pipeline includes:

- A StandardScaler fitted to that group's features
- The best model (XGBoost, Ridge, or Gradient Boosting) retrained on all years ≤ 2023
- A storage mechanism (final_group_models) to support future inference

This setup enables clean and consistent predictions moving into the next phase: using future macroeconomic indicators to forecast GDP.

6.3 Defining Future Feature Values for GDP Prediction

While our models are trained and tuned, they cannot predict future GDP values without corresponding macroeconomic input features for the years ahead. Unfortunately, real forecasts for all required indicators (e.g., exports, inflation, foreign investment) are not always available through public sources like the World Bank or IMF.

To overcome this limitation, we generated synthetic projections of these features using simple and interpretable time-series heuristics. These approximations are not intended to be precise forecasts but rather to enable comparative scenario modeling.

Feature Forecasting Methods

For each country group (G7, BRICS, Other), we estimated future values (2024–2030) using the following approaches:

- Population
  → Projected using average linear growth from the last 5 years per country.

```python
def forecast_population_growth(df, years_to_predict):
    forecasts = []
    for country in df['Country'].unique():
        country_df = df[df['Country'] == country].sort_values('Year')
        pop_values = country_df['Population'].values
        year_values = country_df['Year'].values

        if len(pop_values) < 5:
            continue

        avg_growth = np.diff(pop_values[-5:]).mean()
        last_year = year_values[-1]
        last_value = pop_values[-1]

        for year in years_to_predict:
            y_pred = last_value + avg_growth * (year - last_year)
            forecasts.append({'Country': country, 'Year': year, 'Population': max(0, round(y_pred))})
    return pd.DataFrame(forecasts)
```

- Export & Foreign Direct Investment
  → Forecasted via average annual growth rate, based on the last 5 known years.

```python
def forecast_growth(df, feature, years_to_predict, n_years=5):
    forecasts = []
    for country in df['Country'].unique():
        df_country = df[df['Country'] == country].sort_values('Year')
        y = df_country[feature].dropna().values
        x = df_country['Year'].values

        if len(y) < n_years:
            continue

        avg_growth = np.diff(y[-n_years:]).mean()
        last_year = x[-1]
        last_value = y[-1]

        for year in years_to_predict:
            y_pred = last_value + avg_growth * (year - last_year)
            forecasts.append({'Country': country, 'Year': year, feature: max(0, y_pred)})
    return pd.DataFrame(forecasts)

✓ [299] < 10 ms
```

- Inflation, Unemployment, Terms of Trade
  → Assumed to remain stable, represented by the median value from the last 5 years.

```python
def forecast_flat_median(df, feature, years_to_predict, n_years=5):
    forecasts = []
    for country in df['Country'].unique():
        country_df = df[df['Country'] == country].sort_values('Year')
        y = country_df[feature].dropna().values

        if len(y) < 2:
            continue

        median_val = np.median(y[-n_years:])
        for year in years_to_predict:
            forecasts.append({'Country': country, 'Year': year, feature: median_val})
    return pd.DataFrame(forecasts)

✓ [298] < 10 ms
```

These methods were applied group-wise to ensure internal consistency. Missing or insufficient historical data was handled with fallback logic (e.g., minimum 2 years for median, 5 years for growth).

Constructing Final Feature Set for GDP Forecasting

After forecasting the core macroeconomic indicators, we merged them into a single dataset per group and generated additional derived features to match the structure used during model training.

Merging Forecasted Features

```python
future_years = np.arange(2024, 2031)
future_features = {}

for group in ['G7', 'BRICS', 'Other']:
    group_df = df[df['Group'] == group]

    pop_future          = forecast_population_growth(group_df[['Country', 'Year', 'Population']], future_years)
    export_future       = forecast_growth(group_df[['Country', 'Year', 'Export']], 'Export', future_years)
    fdi_future          = forecast_growth(group_df[['Country', 'Year', 'Foreign Investment']], 'Foreign Investment', future_years)
    terms_future        = forecast_flat_median(group_df[['Country', 'Year', 'TermsOfTrade']], 'TermsOfTrade', future_years)
    inflation_future    = forecast_flat_median(group_df[['Country', 'Year', 'Inflation']], 'Inflation', future_years)
    unemployment_future = forecast_flat_median(group_df[['Country', 'Year', 'Unemployment']], 'Unemployment', future_years)

    future_df = pop_future \
        .merge(export_future, on=['Country', 'Year'], how='outer') \
        .merge(fdi_future, on=['Country', 'Year'], how='outer') \
        .merge(terms_future, on=['Country', 'Year'], how='outer') \
        .merge(inflation_future, on=['Country', 'Year'], how='outer') \
        .merge(unemployment_future, on=['Country', 'Year'], how='outer')

    future_features[group] = future_df
```

For each group (G7, BRICS, Other), we obtained synthetic forecasts for six key features (population, export, foreign investment, inflation, unemployment, and terms of trade).

These forecasts were then merged using an outer join on Country and Year, ensuring that all available future values for each country were included, even if some indicators were partially missing.

This resulted in a complete feature frame for the years 2024–2030, matching the format of the training data.

6.4 Finalizing Future Predictions and Feature Completion

To produce future GDP forecasts (2024–2030), we needed a complete set of input features for each country group (G7, BRICS, Other). However, some required features depend directly on GDP predictions (e.g., GDP growth rate and export-to-GDP ratio). This created a circular dependency, as these features could not be calculated before predicting GDP itself.

To solve this issue, we took the following step-by-step approach:

1. Initial Prediction (Using Available Features)

```
feature_cols = [
    'Population', 'Inflation', 'Unemployment', 'Export', 'Foreign Investment',
    'TermsOfTrade', 'GDP_per_capita','GDP_growth', 'Population_growth', 'Export_to_GDP', 'PCA_1'
]

for group in ['G7', 'BRICS', 'Other']:
    f = future_features[group]

    for col in feature_cols:
        if col not in f.columns:
            f[col] = 0

    X_future = f[feature_cols]
    scaler = final_group_scalers[group]
    X_future_scaled = scaler.transform(X_future)
    model = final_group_models[group]
    y_pred = model.predict(X_future_scaled)
    f['Predicted_GDP'] = y_pred
```

Initially, we ensured each future dataset contained all required columns. Any temporarily missing features (e.g., GDP_per_capita, GDP_growth, PCA_1) were filled with zeros to maintain consistency and shape compatibility.

We then scaled these initial datasets and generated a preliminary GDP prediction. These initial predictions provided a baseline to compute GDP-dependent features

```python
for group in ['G7', 'BRICS', 'Other']:
    f = future_features[group]

    for col in feature_cols:
        if col not in f.columns:
            f[col] = 0

    X_future = f[feature_cols]
    scaler = final_group_scalers[group]
    model = final_group_models[group]
    X_future_scaled = scaler.transform(X_future)
    y_pred = model.predict(X_future_scaled)
    f['Predicted_GDP'] = y_pred

    f['Export_to_GDP'] = f['Export'] / f['Predicted_GDP']
    f['GDP_growth'] = f.groupby('Country')['Predicted_GDP'].pct_change()
    f['Population_growth'] = f.groupby('Country')['Population'].pct_change()
    f[['Export_to_GDP', 'GDP_growth', 'Population_growth']] = f[['Export_to_GDP', 'GDP_growth', 'Population_growth']].fillna(0)

    pca_features = [
        'Population', 'Inflation', 'Unemployment', 'Export',
        'Foreign Investment', 'TermsOfTrade',
        'GDP_growth', 'Population_growth', 'Export_to_GDP'
    ]
    f[pca_features] = f[pca_features].fillna(0)
    X_pca_scaled = scaler_pca.transform(f[pca_features])
    f['PCA_1'] = pca.transform(X_pca_scaled)[:, 0]

    X_future_final = f[feature_cols]
    X_future_final_scaled = scaler.transform(X_future_final)
    y_pred_final = model.predict(X_future_final_scaled)
    f['Predicted_GDP'] = y_pred_final

    future_features[group] = f

    print(f"✅ Ready for {group}: {len(f)} rows, first year: {f['Year'].min()}")

future_all = pd.concat([future_features[group] for group in ['G7', 'BRICS', 'Other']], ignore_index=True)
```

2. Compute GDP-Dependent Features

With the initial GDP predictions, we could now correctly compute the derived features:

- Export-to-GDP ratio: calculated as Export ÷ Predicted GDP
- GDP growth rate: computed via the percent change (pct_change) of predicted GDP over time
- Population growth rate: similarly computed from projected population values

These newly derived features replaced their initial zero placeholders.

3. Recalculating PCA Component

After adding the GDP-dependent features, we recomputed the PCA feature (PCA_1) using the original PCA model trained earlier. This ensured feature consistency with the historical data pipeline.

4. Final Model Prediction

With the complete, recalculated set of features, we scaled the data again and reran the models to obtain the final GDP forecasts.

5. Combining Predictions

```
df_history = df[df['Year'] <= 2023].copy()
df_history['Predicted_GDP'] = df_history['GDP']

full_results = pd.concat([df_history, future_all], ignore_index=True)
```

Finally, we combined all group-level predictions (G7, BRICS, Other) into a unified dataset (future_all) covering all forecast years and countries.

Visualization of GDP Dynamics (2000–2030) for Top-10 Countries

The plot above illustrates historical and projected GDP trends for the ten largest economies from 2000 to 2030, combining actual historical data (2000–2023) and our model-generated forecasts (2024–2030).

Key Observations:

- United States consistently maintains the highest GDP, showing strong growth historically and continuing steady upward trends through 2030.
- China demonstrates rapid historical growth, notably accelerating around 2010–2020. Its predicted GDP continues an upward trajectory but at a more moderate pace post-2025.

- Japan, Germany, and the United Kingdom display more stable growth patterns, indicating developed, mature economies with moderate, consistent economic expansions.
- India exhibits notable projected growth after 2023, suggesting an accelerating economic momentum in the coming years.
- Smaller advanced economies like France, Italy, and Canada have relatively flat trajectories, indicating stable but slow economic growth.
- Brazil shows fluctuations historically, with predictions suggesting minimal or declining growth, possibly reflecting structural economic challenges.

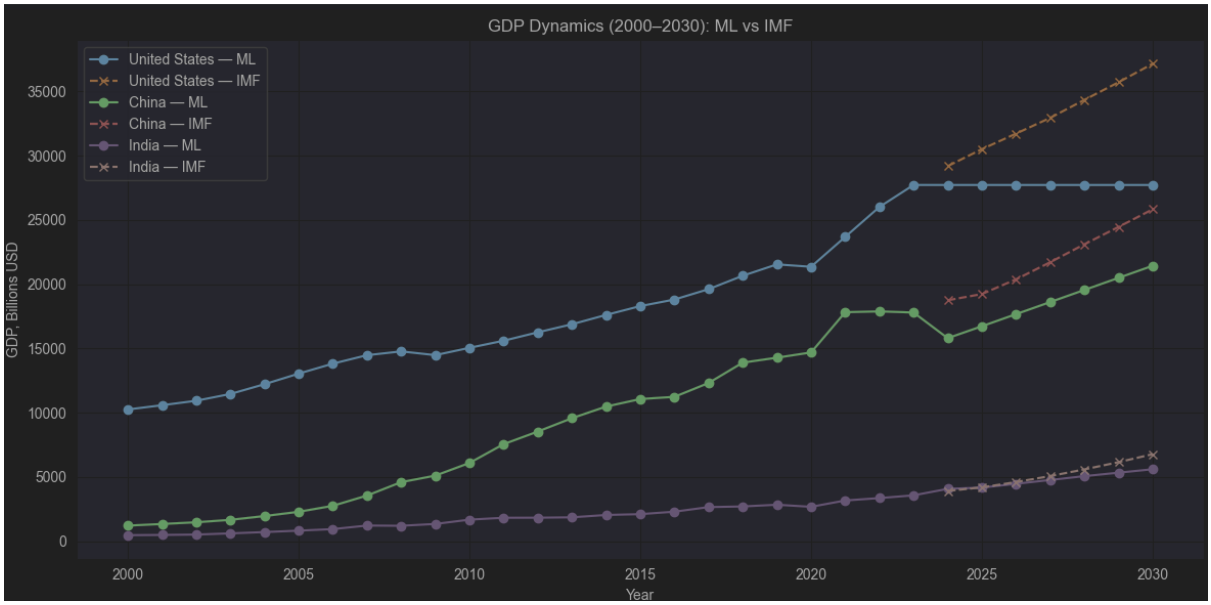
Economic Interpretation:

- Developed economies (G7 countries) display steady, predictable growth patterns, characterized by demographic stability and mature market structures.
- Emerging economies (China, India) continue to grow robustly but with slower incremental growth rates over time, reflecting economic maturation and changing demographic dynamics.
- The diverse dynamics among these top economies underline structural differences such as demographic trends, export-driven growth strategies, and market maturity.

This visualization confirms the model's capability to extend historical economic trends realistically, highlighting distinct growth patterns across different economic groups.

6.5 Comparison of ML and IMF GDP Forecasts: Key Findings

Comparing our machine learning-based GDP forecasts with official IMF projections (2025, 2027, 2030) highlights important similarities as well as clear distinctions in how the two methods predict future economic dynamics.



Our ML approach, utilizing group-specific models tailored to the characteristics of G7, BRICS, and Other economies, produces purely data-driven forecasts based on historical macroeconomic relationships. In contrast, IMF forecasts integrate expert judgment, future policy scenarios, and anticipated global economic shifts that historical data alone cannot fully capture.

Emerging Economies (China, India)

- Our ML model closely aligns with IMF forecasts, capturing the strong upward growth trajectory.
- This indicates that historical macroeconomic trends, especially exports, investment, and population dynamics, provide robust predictive power for these fast-growing economies.

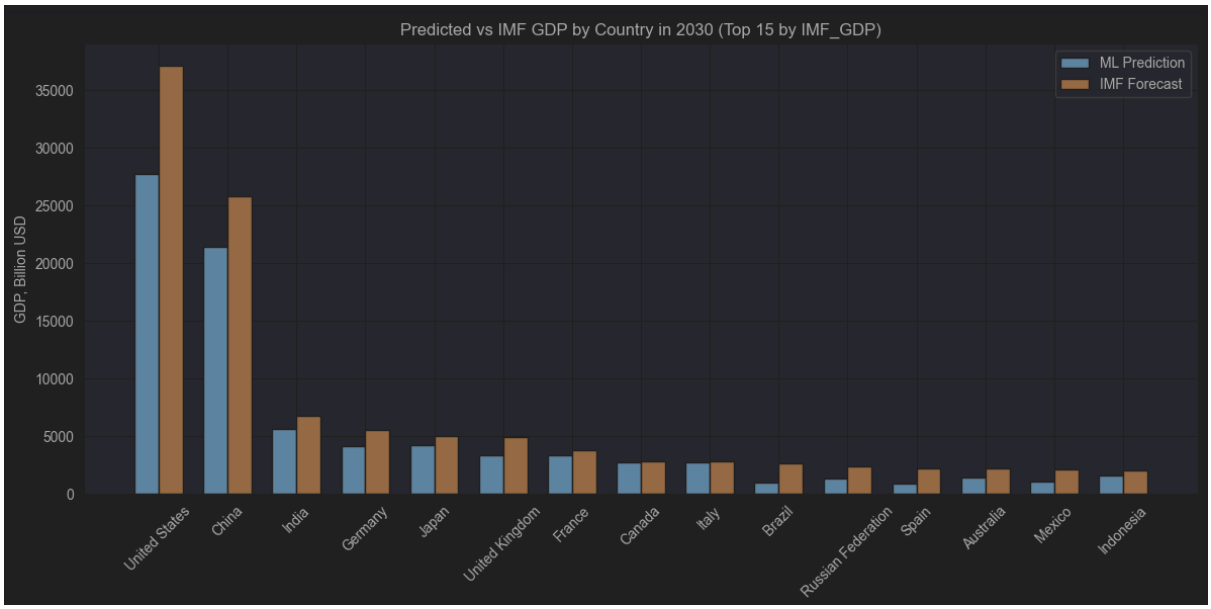
Advanced Economies (United States, Germany, etc.)

- For advanced economies, especially the United States, our ML model forecasts lower GDP values than the IMF.
- The model shows more conservative predictions, largely driven by stable historical relationships, without incorporating future innovations, policy adjustments, or technological breakthroughs expected by IMF experts.

- Consequently, while IMF forecasts project continuous growth, ML forecasts suggest slower growth or stabilization in GDP for mature economies due to the absence of strong upward macroeconomic indicators in historical trends.

Intermediate Economies (Brazil, Russia)

- ML forecasts for economies facing structural or political challenges tend to diverge more noticeably from IMF projections.
- Differences arise because IMF scenarios account explicitly for future structural reforms, potential policy improvements, or shifts that purely historical data-driven models cannot anticipate.



Conclusion and Implications

This analysis emphasizes both the strengths and limitations of purely historical machine learning forecasts:

- ML models excel at providing objective, transparent forecasts grounded in historical economic patterns, particularly effective in stable-growth or rapidly developing scenarios.
- However, they inherently miss future external shocks, policy innovations, and transformative technological or structural changes.
- A comprehensive approach to GDP forecasting should ideally integrate ML-driven, data-based insights with expert-driven IMF-style projections to create robust, scenario-aware economic outlooks.

Thus, our model serves as a powerful complement to traditional forecasting methods, providing a rigorous baseline from which scenario analyses and policy-based forecasts can be effectively constructed

7. Conclusion

This project resulted in a comprehensive analysis and forecasting of global gross domestic product (GDP) based on macroeconomic indicators for the period from 2000 to 2023. We used a powerful analytical approach covering several key stages: data mining (EDA), preprocessing, feature generation, modeling, and hyperparameter tuning.

Key findings of the project:

1. Dividing countries into economic groups (G7, BRICS, and Other) turned out to be a critical decision. It significantly improved the quality of forecasting, allowing models to better capture the specific economic patterns of each group.

2. The most important macroeconomic indicators that determine GDP differ significantly across groups:

- G7: Population size turned out to be the most important feature, which confirms the hypothesis of maturity and demographic stability of the leading economies.
- BRICS: Here, the GDP forecast is heavily weighted towards exports and investment, with a high value for the synthetic component (PCA) reflecting the economic structure of the countries in this group.
- Other: The absolute dominance of the export indicator highlights the importance of foreign trade for smaller and more diverse economies.

3. Machine learning models were effective in forecasting GDP, but their results differed significantly from the IMF forecasts:

- For fast-growing economies (China, India), the ML model and the IMF were close, confirming the strength of historical trends and data.
- For developed countries (USA, Germany, Japan), our ML model was more conservative, forecasting more subdued GDP growth rates, as it did not take into account future changes in economic policy and innovation, which the IMF focuses on.
- For economies with volatility and structural challenges (Brazil, Russia), the ML model also differed significantly from the IMF forecasts, which emphasizes the need for expert judgment when forecasting economies with high uncertainty.

4. Hyperparameter tuning did not always produce a positive effect:

- For the G7 and BRICS, the best results were shown by non-tuned versions of the models (XGBoost and Ridge), indicating the need for a cautious approach to model complexity.
- For the Other group, tuning the Gradient Boosting model was clearly justified, providing a noticeable improvement in metrics.

Economic observations and recommendations:

- The results emphasized the importance of trade activity (exports) as the leading driver of GDP growth in developing and mixed economies.
- In contrast, for developed countries, the demographic factor becomes key, indicating the need to take into account social and demographic policies in long-term GDP forecasting.
- The PCA indicator has proven its value as a general economic indicator, successfully compressing information from many features into a single component.

Methodological conclusions and further development of the project:

- This project confirmed that purely historical data can provide accurate and robust forecasts for countries with stable or clearly defined economic trends.
- However, the project revealed the limitations of this approach, especially for forecasting economies with high uncertainty or when taking into account future economic innovations.
- To improve the accuracy of forecasting in the future, it is recommended to integrate machine learning with expert scenario forecasts, such as those of the IMF, to create a comprehensive and realistic forecast model.

Thus, the project achieved the main goal - to demonstrate the capabilities and limitations of data-oriented methods for GDP forecasting, identify the most important economic drivers, and provide a deep understanding of how various economic groups are developing and will develop in the coming years. The results of this work can serve as a basis for making informed economic decisions and planning at the global level.