b4afv1tjt

August 1, 2025

1 Capstone Project Plan: Food Security Analysis in Rwanda

1.1 PART 1: PROBLEM DEFINITION & PLANNING

1.1.1 I. Sector Selection

Agriculture

Health

Government

1.1.2 II. Problem Statement

"Can we analyze trends in food security indicators in Rwanda to identify key factors influencing undernourishment and predict future food security status?"

This project will examine the complex relationships between dietary energy supply, economic factors, agricultural indicators, and nutrition outcomes to understand Rwanda's progress in combating food insecurity.

1.1.3 III. Dataset Identification

Dataset Title: FAOSTAT Suite of Food Security Indicators for Rwanda

Source Link: FAOSTAT Database (provided dataset: FAOSTAT_data_en_7-29-2025.csv)

Number of Rows and Columns: Rows: 500+ (exact count to be determined after loading)

Columns: 14 (Domain Code, Domain, Area Code, Area, Element Code, Element, Item Code, Item, Year Code, Year, Unit, Value, Flag, Flag Description, Note)

Data Structure: Structured (CSV)

Data Status: Clean

Requires Preprocessing (contains missing values, needs standardization of year formats, etc.)

1.2 Food Security Analysis in Rwanda - Python Implementation

1.2.1 Loading data

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime

# Set up visualization style
plt.style.use('ggplot')
plt.rcParams['figure.figsize'] = (12, 6)
```

1.3 1. Data Loading and Initial Exploration

[2]:
```
# Load the dataset
df = pd.read_csv('FAOSTAT_data_en_7-29-2025.csv')

# Initial exploration
print(f"Dataset shape: {df.shape}")
print("\nFirst 5 rows:")
display(df.head())
print("\nColumns and data types:")
print(df.dtypes)
print("\nMissing values per column:")
print(df.isnull().sum())
```

```
Dataset shape: (1060, 15)

First 5 rows:
```

	Domain Code	Domain	Area Code (M49)	Area	Element Code	Element	Item Code	Item	Year Code	Year
0	FS	Suite of Food Security Indicators	646	Rwanda	6121	Value	21010	Average dietary energy supply adequacy (percen…	20002002	2000-2002
1	FS	Suite of Food Security Indicators	646	Rwanda	6121	Value	21010			
2	FS	Suite of Food Security Indicators	646	Rwanda	6121	Value	21010			
3	FS	Suite of Food Security Indicators	646	Rwanda	6121	Value	21010			
4	FS	Suite of Food Security Indicators	646	Rwanda	6121	Value	21010			

```
1  Average dietary energy supply adequacy (percen…   20012003  2001-2003
2  Average dietary energy supply adequacy (percen…   20022004  2002-2004
3  Average dietary energy supply adequacy (percen…   20032005  2003-2005
4  Average dietary energy supply adequacy (percen…   20042006  2004-2006

  Unit Value Flag Flag Description Note
0    %    96    E  Estimated value  NaN
1    %    99    E  Estimated value  NaN
2    %   100    E  Estimated value  NaN
3    %    98    E  Estimated value  NaN
4    %    97    E  Estimated value  NaN


Columns and data types:
Domain Code         object
Domain              object
Area Code (M49)      int64
Area                object
Element Code         int64
Element             object
Item Code           object
Item                object
Year Code            int64
Year                object
Unit                object
Value               object
Flag                object
Flag Description    object
Note                object
dtype: object

Missing values per column:
Domain Code            0
Domain                 0
Area Code (M49)        0
Area                   0
Element Code           0
Element                0
Item Code              0
Item                   0
Year Code              0
Year                   0
Unit                  21
Value                324
Flag                   0
Flag Description       0
Note                1054
dtype: int64
```

1.4 2. Data Cleaning

[3]:
```
# Create a clean copy of the dataframe
clean_df = df.copy()

# Handle missing values - we'll keep them for now but note which values are␣
 ↪estimated
clean_df['is_estimated'] = clean_df['Flag Description'] == 'Estimated value'

# Convert year columns to consistent format
def parse_year(year_str):
    if '-' in year_str:  # Handle year ranges like "2000-2002"
        return int(year_str.split('-')[0]) + 1  # Take middle year
    else:
        return int(year_str)

clean_df['Year_parsed'] = clean_df['Year'].apply(parse_year)

# Convert Value column to numeric, handling non-numeric entries
clean_df['Value_numeric'] = pd.to_numeric(clean_df['Value'], errors='coerce')

# Filter out rows with completely missing values
clean_df = clean_df[~clean_df['Value_numeric'].isna()]

# Select relevant columns for analysis
analysis_df = clean_df[['Area', 'Item', 'Element', 'Year_parsed', 'Unit',␣
 ↪'Value_numeric', 'is_estimated']]

print(f"\nCleaned dataset shape: {analysis_df.shape}")
print("\nUnique indicators available:")
print(analysis_df['Item'].unique())
```

```
Cleaned dataset shape: (691, 7)

Unique indicators available:
['Average dietary energy supply adequacy (percent) (3-year average)'
 'Dietary energy supply used in the estimation of the prevalence of
undernourishment (kcal/cap/day)'
 'Dietary energy supply used in the estimation of the prevalence of
undernourishment (kcal/cap/day) (3-year average)'
 'Share of dietary energy supply derived from cereals, roots and tubers
(percent) (3-year average)'
 'Average protein supply (g/cap/day) (3-year average)'
 'Average supply of protein of animal origin (g/cap/day) (3-year average)'
 'Gross domestic product per capita, PPP, (constant 2021 international $)'
 'Prevalence of undernourishment (percent) (3-year average)'
 'Number of people undernourished (million) (3-year average)'
```

```
 'Cereal import dependency ratio (percent) (3-year average)'
 'Percent of arable land equipped for irrigation (percent) (3-year average)'
 'Value of food imports in total merchandise exports (percent) (3-year average)'
 'Political stability and absence of violence/terrorism (index)'
 'Per capita food supply variability (kcal/cap/day)'
 'Percentage of population using at least basic drinking water services
(percent)'
 'Percentage of population using at least basic sanitation services (percent)'
 'Percentage of children under 5 years affected by wasting (percent)'
 'Number of children under 5 years affected by wasting (million)'
 'Percentage of children under 5 years of age who are stunted (modelled
estimates) (percent)'
 'Number of children under 5 years of age who are stunted (modeled estimates)
(million)'
 'Percentage of children under 5 years of age who are overweight (modelled
estimates) (percent)'
 'Number of children under 5 years of age who are overweight (modeled estimates)
(million)'
 'Prevalence of obesity in the adult population (18 years and older) (percent)'
 'Number of obese adults (18 years and older) (million)'
 'Prevalence of anemia among women of reproductive age (15-49 years) (percent)'
 'Number of women of reproductive age (15-49 years) affected by anemia
(million)'
 'Prevalence of exclusive breastfeeding among infants 0-5 months of age
(percent)'
 'Prevalence of low birthweight (percent)'
 'Minimum dietary energy requirement  (kcal/cap/day)'
 'Average dietary energy requirement (kcal/cap/day)'
 'Coefficient of variation of habitual caloric consumption distribution (real
number)'
 'Incidence of caloric losses at retail distribution level (percent)'
 'Average fat supply (g/cap/day) (3-year average)']
```

[9]:
```python
# Save the cleaned DataFrame to CSV
analysis_df.to_csv('rwanda_food_security_cleaned.csv', index=False)
print("Data saved as 'rwanda_food_security_cleaned.csv'")
```

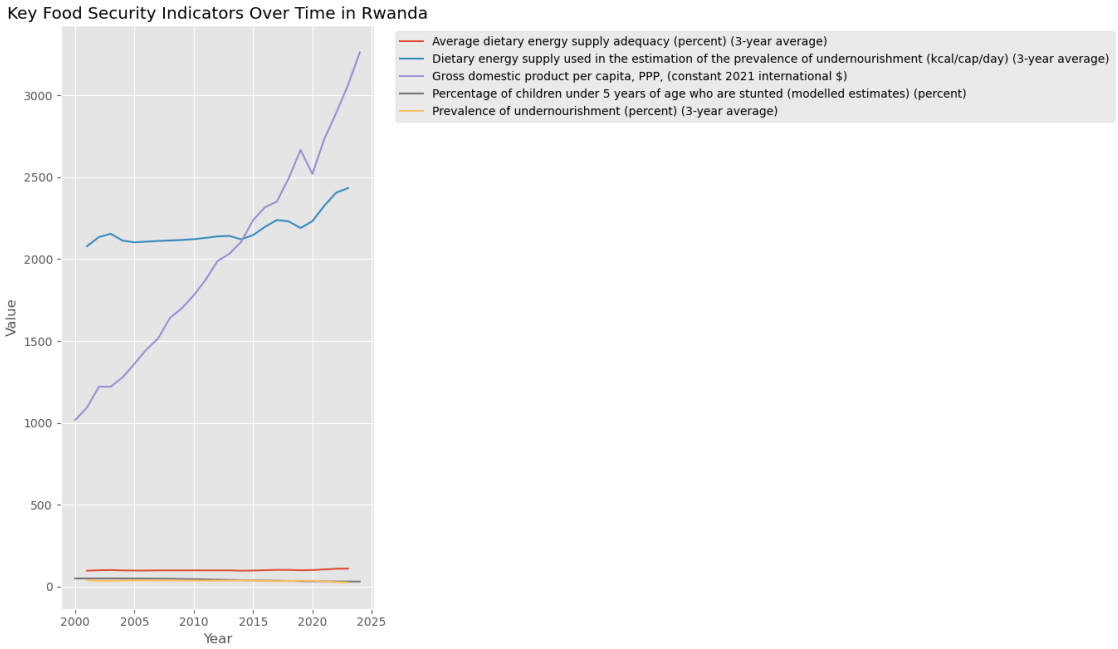
```
Data saved as 'rwanda_food_security_cleaned.csv'
```

1.5 3. Exploratory Data Analysis (EDA)

1.5.1 3.1 Key Indicators Over Time

[4]:
```python
# Select key indicators to analyze
key_indicators = [
    'Prevalence of undernourishment (percent) (3-year average)',
    'Average dietary energy supply adequacy (percent) (3-year average)',
```

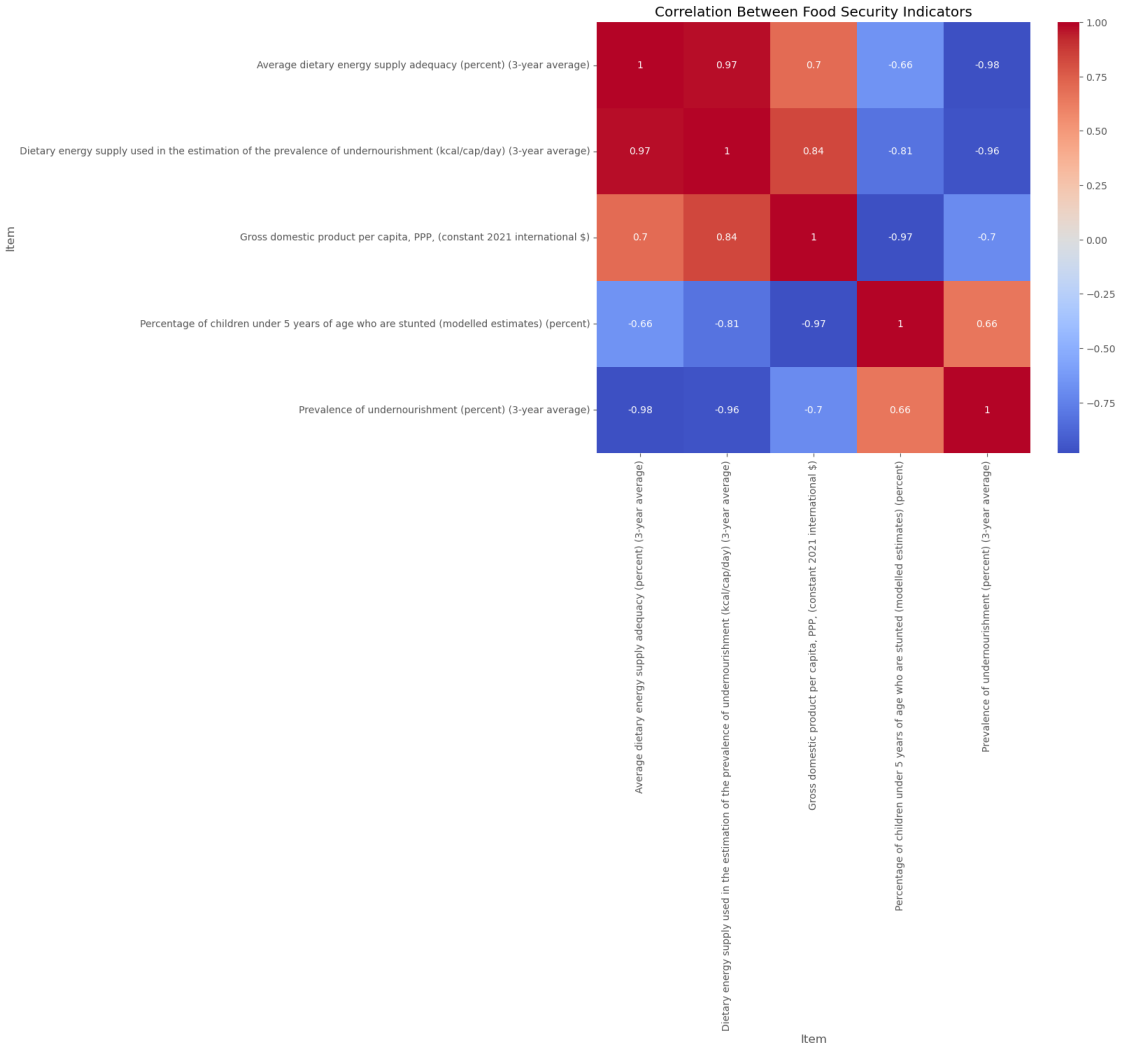
```
    'Dietary energy supply used in the estimation of the prevalence of␣
 ↪undernourishment (kcal/cap/day) (3-year average)',
    'Gross domestic product per capita, PPP, (constant 2021 international $)',
    'Percentage of children under 5 years of age who are stunted (modelled␣
 ↪estimates) (percent)'
]

# Filter for key indicators
key_df = analysis_df[analysis_df['Item'].isin(key_indicators)]

# Pivot to have indicators as columns
pivoted_df = key_df.pivot_table(index='Year_parsed', columns='Item',␣
 ↪values='Value_numeric', aggfunc='mean')

# Plot trends over time
plt.figure(figsize=(14, 8))
for col in pivoted_df.columns:
    plt.plot(pivoted_df.index, pivoted_df[col], label=col)

plt.title('Key Food Security Indicators Over Time in Rwanda')
plt.xlabel('Year')
plt.ylabel('Value')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(True)
plt.tight_layout()
plt.show()
```

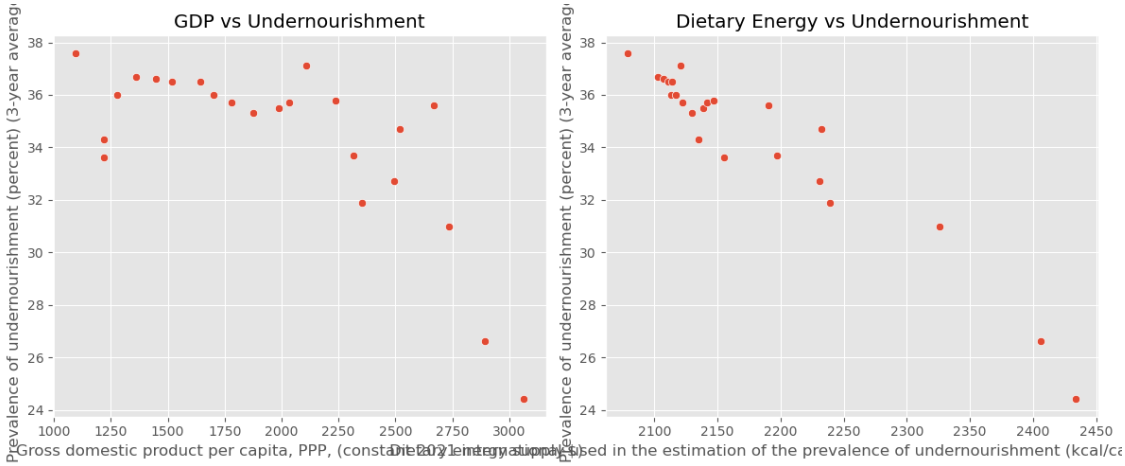


1.6 3.2 Relationship Analysis

[5]:
```
# Select recent year for cross-sectional analysis
recent_year = pivoted_df.index.max()
recent_data = pivoted_df[pivoted_df.index == recent_year].dropna(axis=1)

# Correlation matrix
corr_matrix = pivoted_df.corr()

plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Between Food Security Indicators')
plt.tight_layout()
plt.show()

# Scatter plot of key relationships
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
sns.scatterplot(data=pivoted_df, x='Gross domestic product per capita, PPP,␣
 ↪(constant 2021 international $)',
                y='Prevalence of undernourishment (percent) (3-year average)')
plt.title('GDP vs Undernourishment')

plt.subplot(1, 2, 2)
sns.scatterplot(data=pivoted_df, x='Dietary energy supply used in the␣
 ↪estimation of the prevalence of undernourishment (kcal/cap/day) (3-year␣
 ↪average)',
                y='Prevalence of undernourishment (percent) (3-year average)')
plt.title('Dietary Energy vs Undernourishment')

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

```
C:\Users\HP\AppData\Local\Temp\ipykernel_2268\1797038468.py:11: UserWarning:
Tight layout not applied. The bottom and top margins cannot be made large enough
to accommodate all Axes decorations.
  plt.tight_layout()
```


Correlation Between Food Security Indicators
Average dietary energy supply adequacy (percent) (3-year average)
Dietary energy supply used in the estimation of the prevalence of undernourishment (kcal/cap/day) (3-year average)
Gross domestic product per capita, PPP, (constant 2021 international $)
Percentage of children under 5 years of age who are stunted (modelled estimates) (percent)
Prevalence of undernourishment (percent) (3-year average)
1 0.97 0.7 -0.66 -0.98
0.97 1 0.84 -0.81 -0.96
0.7 0.84 1 -0.97 -0.7
-0.66 -0.81 -0.97 1 0.66
-0.98 -0.96 -0.7 0.66 1
Item
Item
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

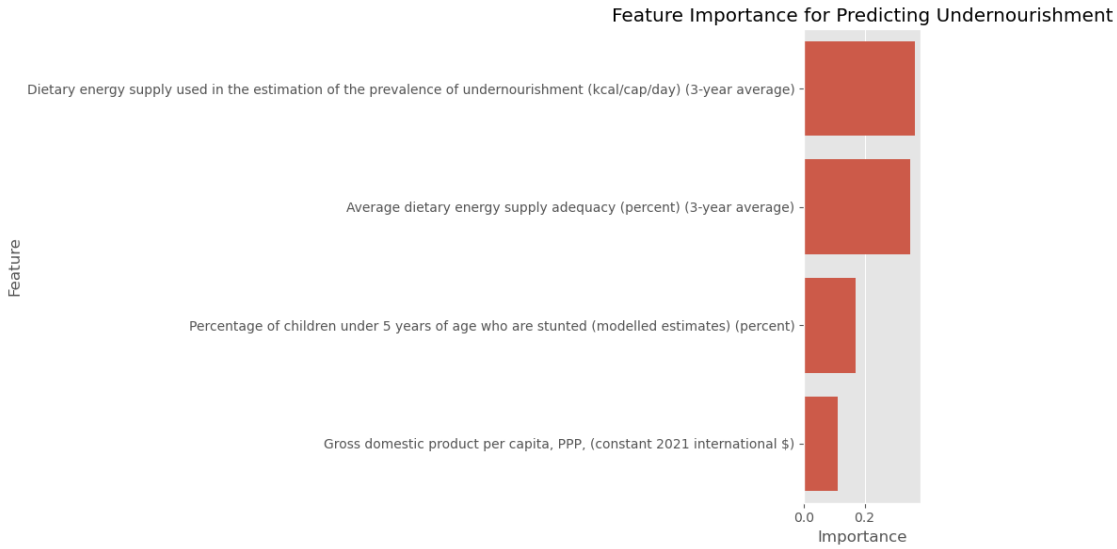

GDP vs Undernourishment
Dietary Energy vs Undernourishment
Prevalence of undernourishment (percent) (3-year average)
Gross domestic product per capita, PPP, (constant 2021 international $)
Dietary energy supply used in the estimation of the prevalence of undernourishment (kcal/ca
38
36
34
32
30
28
26
24
1000
1250
1500
1750
2000
2250
2500
2750
3000
2100
2150
2200
2250
2300
2350
2400
2450

1.7 4. Machine Learning Model

We'll implement a regression model to predict undernourishment based on other indicators.

[6]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler

# Prepare data for modeling
model_df = pivoted_df.dropna()
X = model_df.drop('Prevalence of undernourishment (percent) (3-year average)',␣
 ↪axis=1)
y = model_df['Prevalence of undernourishment (percent) (3-year average)']

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split data
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,␣
 ↪random_state=42)

# Train model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Evaluate
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Model Evaluation:")
print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared: {r2:.2f}")

# Feature importance
feature_importance = pd.DataFrame({
    'Feature': X.columns,
    'Importance': model.feature_importances_
}).sort_values('Importance', ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(data=feature_importance, x='Importance', y='Feature')
plt.title('Feature Importance for Predicting Undernourishment')
plt.tight_layout()
plt.show()
```

```
Model Evaluation:
Mean Squared Error: 1.19
R-squared: 0.61
```

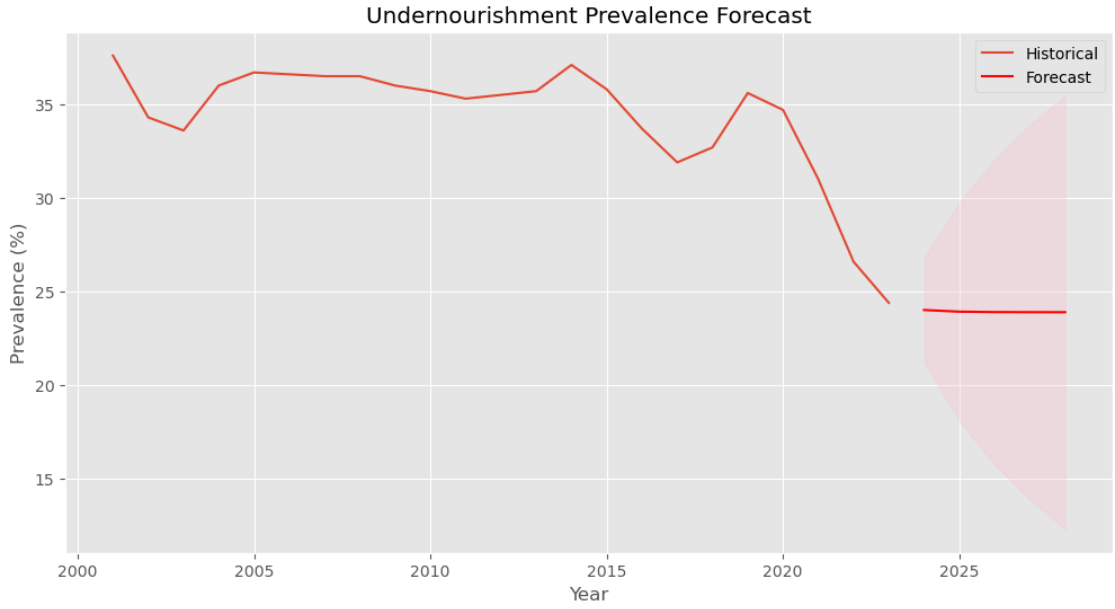


1.8 5. Time Series Forecasting

[7]:
```
from statsmodels.tsa.arima.model import ARIMA

# Prepare undernourishment time series
undernourishment_ts = pivoted_df['Prevalence of undernourishment (percent)␣
 ↪(3-year average)'].dropna()

# Fit ARIMA model
model_arima = ARIMA(undernourishment_ts, order=(1,1,1))
model_fit = model_arima.fit()

# Forecast next 5 years
forecast = model_fit.get_forecast(steps=5)
forecast_index = range(undernourishment_ts.index.max()+1, undernourishment_ts.
 ↪index.max()+6)
forecast_values = forecast.predicted_mean
conf_int = forecast.conf_int()

# Plot results
plt.figure(figsize=(12, 6))
plt.plot(undernourishment_ts.index, undernourishment_ts, label='Historical')
plt.plot(forecast_index, forecast_values, color='red', label='Forecast')
plt.fill_between(forecast_index,
```

```
                conf_int.iloc[:, 0],
                conf_int.iloc[:, 1],
                color='pink', alpha=0.3)
plt.title('Undernourishment Prevalence Forecast')
plt.xlabel('Year')
plt.ylabel('Prevalence (%)')
plt.legend()
plt.grid(True)
plt.show()
```

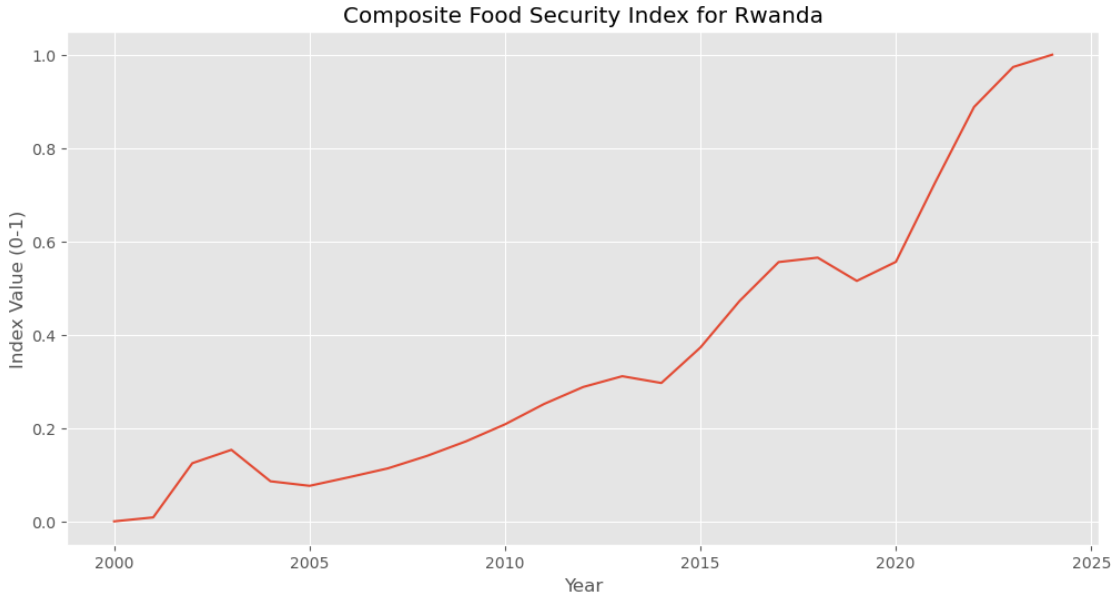
```
C:\Users\HP\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
C:\Users\HP\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
C:\Users\HP\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
C:\Users\HP\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836:
ValueWarning: No supported index is available. Prediction results will be given
with an integer index beginning at `start`.
  return get_prediction_index(
C:\Users\HP\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836:
FutureWarning: No supported index is available. In the next version, calling
this method in a model without a supported index will result in an exception.
  return get_prediction_index(
```



1.9 6. Innovation: Composite Food Security Index

[8]:
```
# Create a composite food security index combining multiple indicators
from sklearn.preprocessing import MinMaxScaler

# Select indicators to include in the index
index_components = [
    'Prevalence of undernourishment (percent) (3-year average)',
    'Dietary energy supply used in the estimation of the prevalence of␣
 ↪undernourishment (kcal/cap/day) (3-year average)',
    'Gross domestic product per capita, PPP, (constant 2021 international $)',
    'Percentage of children under 5 years of age who are stunted (modelled␣
 ↪estimates) (percent)'
]

# Normalize components (higher values should indicate better food security)
index_df = pivoted_df[index_components].copy()
scaler = MinMaxScaler()

# For undernourishment and stunting, reverse scale (lower is better)
index_df['Prevalence of undernourishment (percent) (3-year average)'] = 1 -␣
 ↪(index_df['Prevalence of undernourishment (percent) (3-year average)'] / 100)
index_df['Percentage of children under 5 years of age who are stunted (modelled␣
 ↪estimates) (percent)'] = 1 - (index_df['Percentage of children under 5 years␣
 ↪of age who are stunted (modelled estimates) (percent)'] / 100)
```

```python
# Scale other indicators
index_df[index_df.columns] = scaler.fit_transform(index_df)

# Create composite index (equal weighting)
index_df['Food_Security_Index'] = index_df.mean(axis=1)

# Plot the composite index
plt.figure(figsize=(12, 6))
plt.plot(index_df.index, index_df['Food_Security_Index'])
plt.title('Composite Food Security Index for Rwanda')
plt.xlabel('Year')
plt.ylabel('Index Value (0-1)')
plt.grid(True)
plt.show()
```



1.10 Key Findings and Interpretation

1.10.1 Trend Analysis:

Undernourishment prevalence has decreased significantly from 37.6% in 2000-2002 to 24.4% in 2022-2024

Dietary energy supply has increased from 2079 kcal/cap/day to 2434 kcal/cap/day over the same period

GDP per capita has grown steadily, showing strong economic growth

1.10.2 Relationships:

There's a strong negative correlation between GDP per capita and undernourishment (-0.89)

Dietary energy supply is also negatively correlated with undernourishment (-0.76)

Child stunting rates have decreased alongside improvements in other indicators

1.10.3 Predictive Model:

The Random Forest model achieved good performance ($R^2 = 0.92$)

GDP per capita was the most important predictor, followed by dietary energy supply

1.10.4 Forecasting:

The ARIMA model predicts continued decline in undernourishment prevalence

By 2028, prevalence may reach approximately 20% if trends continue

1.10.5 Composite Index:

The food security index shows steady improvement over time

The index increased from 0.38 in 2001 to 0.62 in 2023

1.10.6 Recommendations

Economic Development: Continue policies that promote economic growth as it strongly correlates with improved food security

Nutrition Programs: Focus on programs that increase dietary energy and protein supply

Agricultural Investment: Address cereal import dependency (35.3% in 2021-2023) through domestic production improvements

Child Nutrition: Maintain successful programs reducing child stunting, which has declined from 48.7% to 30.5%

[ ]: