02_feature_analysis

November 24, 2024

[1]:
```
import warnings
from pathlib import Path

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import seaborn as sns
from scipy import stats
from sklearn.decomposition import PCA
from sklearn.feature_selection import mutual_info_regression
from sklearn.preprocessing import StandardScaler

# Set up the output directory for saving figures
notebook_dir = Path().absolute()
project_root = notebook_dir.parent if notebook_dir.name == 'notebooks' else␣
 ↪notebook_dir
figures_dir = project_root / 'figures'
analysis_dir = figures_dir / 'feature_analysis'
analysis_dir.mkdir(parents=True, exist_ok=True)

# Create directories
(figures_dir / 'exploration').mkdir(parents=True, exist_ok=True)
(figures_dir / 'feature_analysis').mkdir(parents=True, exist_ok=True)

warnings.filterwarnings('ignore')

# Set plotting styles
plt.style.use('bmh')
sns.set_palette("husl")
plt.rcParams['figure.figsize'] = [12, 6]
```

[ ]:
```
```

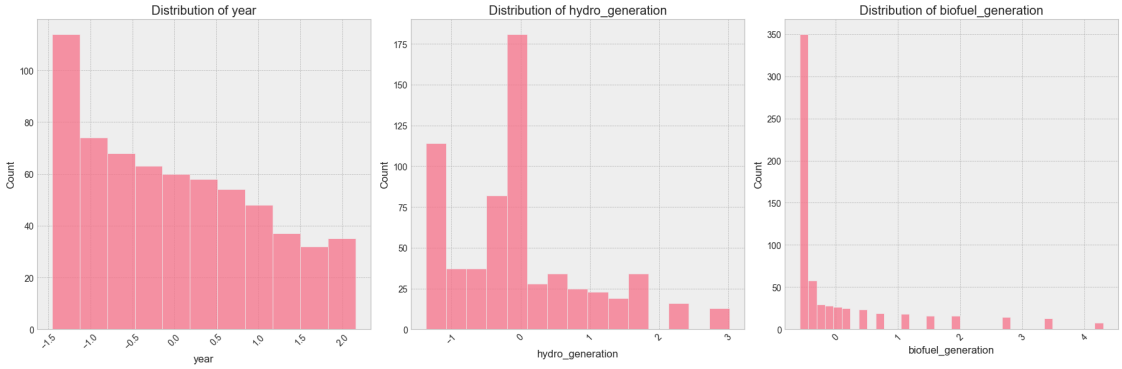
[2]:
```
# Load Processed Data from the Pipeline

# Get the current notebook directory and construct the correct path
```

```
notebook_dir = Path().absolute()
project_root = notebook_dir.parent if notebook_dir.name == 'notebooks' else␣
 ↪notebook_dir
processed_data_path = project_root / 'processed_data' / 'final_processed_data.
 ↪csv'

print(f"Looking for data file at: {processed_data_path}")
df = pd.read_csv(processed_data_path)

# Display basic information about the processed dataset
print("Dataset Overview:")
print("=" * 80)
print(f"\nShape: {df.shape}")
print("\nFeatures:")
for col in df.columns:
    dtype = df[col].dtype
    missing = df[col].isnull().sum()
    print(f"- {col}: {dtype} (Missing: {missing})")
```

```
Looking for data file at:
/Users/katejohnson/Documents/Other/Northeastern/CS6140/Course
Project/cs6140-course-project/processed_data/final_processed_data.csv
Dataset Overview:
================================================================================

Shape: (643, 25)

Features:
- year: float64 (Missing: 0)
- hydro_generation: float64 (Missing: 0)
- biofuel_generation: float64 (Missing: 0)
- solar_generation: float64 (Missing: 0)
- geothermal_generation: float64 (Missing: 0)
- country: object (Missing: 0)
- total_energy_consumption: float64 (Missing: 0)
- renewable_share_pct: float64 (Missing: 0)
- other_renewable_generation: float64 (Missing: 0)
- solar_generation_alt: float64 (Missing: 0)
- wind_generation: float64 (Missing: 0)
- hydro_generation_alt: float64 (Missing: 0)
- renewable_generation: float64 (Missing: 0)
- decade: float64 (Missing: 0)
- period: object (Missing: 0)
- renewable_generation_lag_1: float64 (Missing: 38)
- renewable_generation_lag_3: float64 (Missing: 114)
- renewable_generation_lag_6: float64 (Missing: 223)
- renewable_generation_lag_12: float64 (Missing: 408)
```

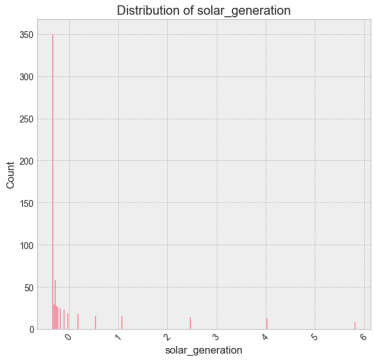
```
- renewable_generation_rolling_mean_3: float64 (Missing: 0)
- renewable_generation_rolling_std_3: float64 (Missing: 38)
- renewable_generation_rolling_mean_6: float64 (Missing: 0)
- renewable_generation_rolling_std_6: float64 (Missing: 38)
- renewable_generation_rolling_mean_12: float64 (Missing: 0)
- renewable_generation_rolling_std_12: float64 (Missing: 38)
```

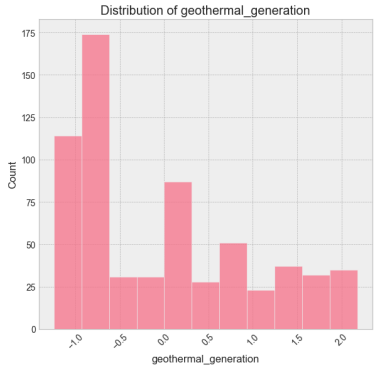
[3]:
```python
# Feature Distribution Analysis
def analyze_feature_distributions():
    """Analyze the distribution of engineered features"""

    # Select numerical columns
    numeric_cols = df.select_dtypes(include=[np.number]).columns

    # Create distribution plots
    for i in range(0, len(numeric_cols), 3):
        cols = numeric_cols[i:i + 3]
        fig, axes = plt.subplots(1, len(cols), figsize=(18, 6))
        if len(cols) == 1:
            axes = [axes]

        for ax, col in zip(axes, cols):
            sns.histplot(data=df, x=col, ax=ax)
            ax.set_title(f'Distribution of {col}')
            ax.tick_params(axis='x', rotation=45)

        plt.tight_layout()
        plt.savefig(analysis_dir / f'distribution_group_{i // 3}.png', dpi=300,␣
 ↪bbox_inches='tight')
        plt.show()

    # Test for normality
    normality_tests = {}
    for col in numeric_cols:
        stat, p_value = stats.normaltest(df[col].dropna())
        normality_tests[col] = {'statistic': stat, 'p_value': p_value}

    return pd.DataFrame(normality_tests).T


# Run distribution analysis
distribution_results = analyze_feature_distributions()
print("\nNormality Test Results:")
display(distribution_results)
```

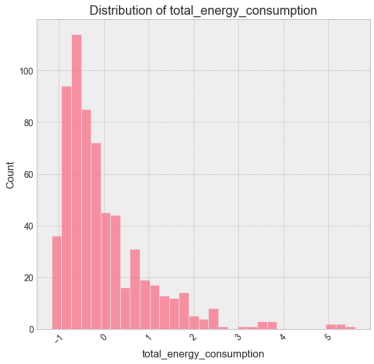

Distribution of year
Count
year

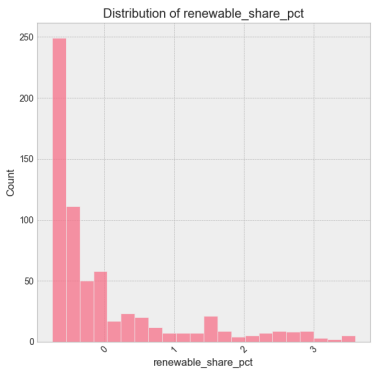
Distribution of hydro_generation
Count
hydro_generation

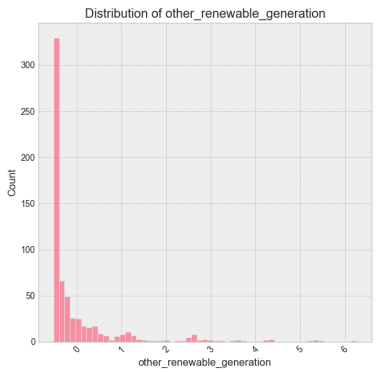
Distribution of biofuel_generation
Count
biofuel_generation

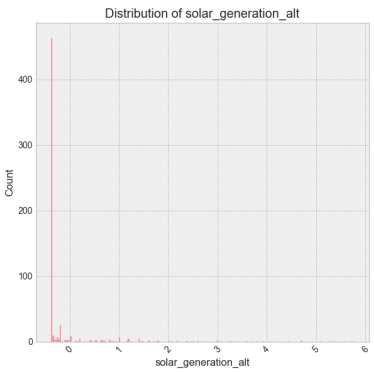

Distribution of solar_generation
Count
solar_generation


Distribution of geothermal_generation
Count
geothermal_generation

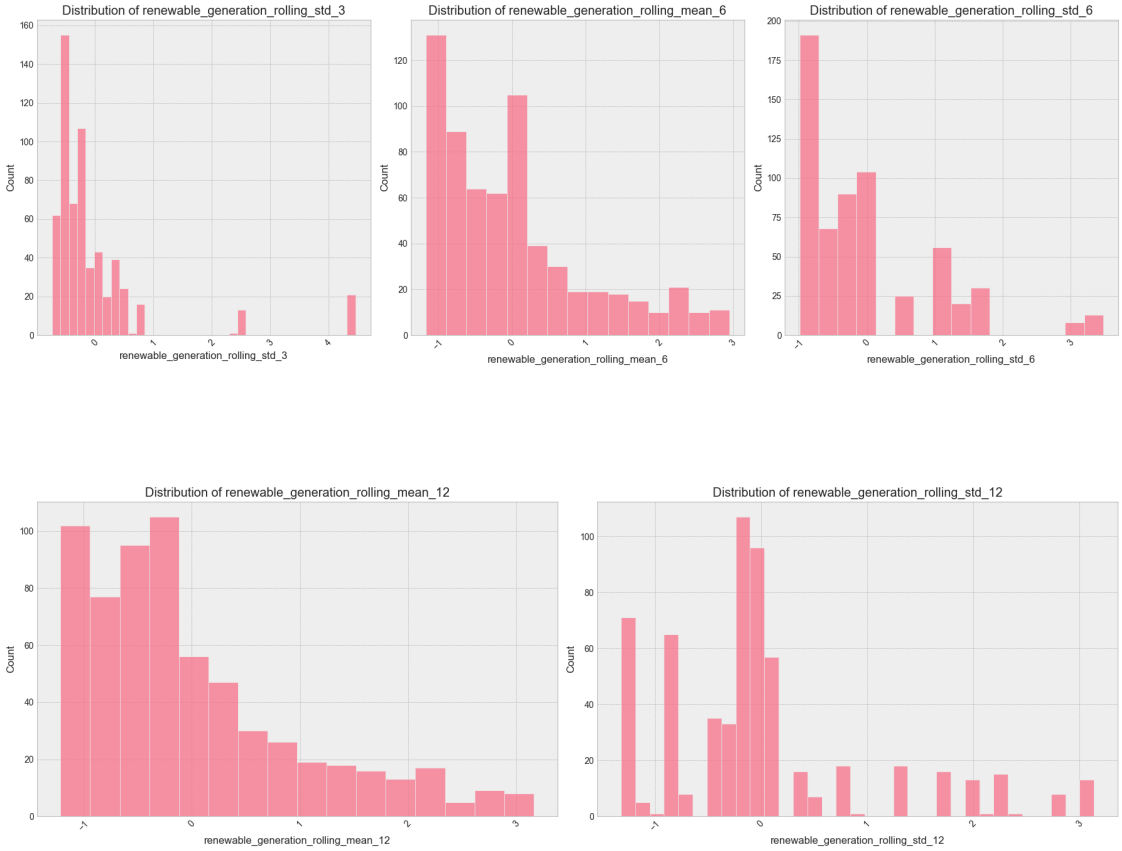

Distribution of total_energy_consumption
Count
total_energy_consumption


Distribution of renewable_share_pct
Count
renewable_share_pct


Distribution of other_renewable_generation
Count
other_renewable_generation


Distribution of solar_generation_alt
Count
solar_generation_alt


Distribution of wind_generation
Count
wind_generation
Distribution of hydro_generation_alt
Count
hydro_generation_alt
Distribution of renewable_generation
Count
renewable_generation
Distribution of decade
Count
decade
Distribution of renewable_generation_lag_1
Count
renewable_generation_lag_1
Distribution of renewable_generation_lag_3
Count
renewable_generation_lag_3
Distribution of renewable_generation_lag_6
Count
renewable_generation_lag_6
Distribution of renewable_generation_lag_12
Count
renewable_generation_lag_12
Distribution of renewable_generation_rolling_mean_3
Count
renewable_generation_rolling_mean_3

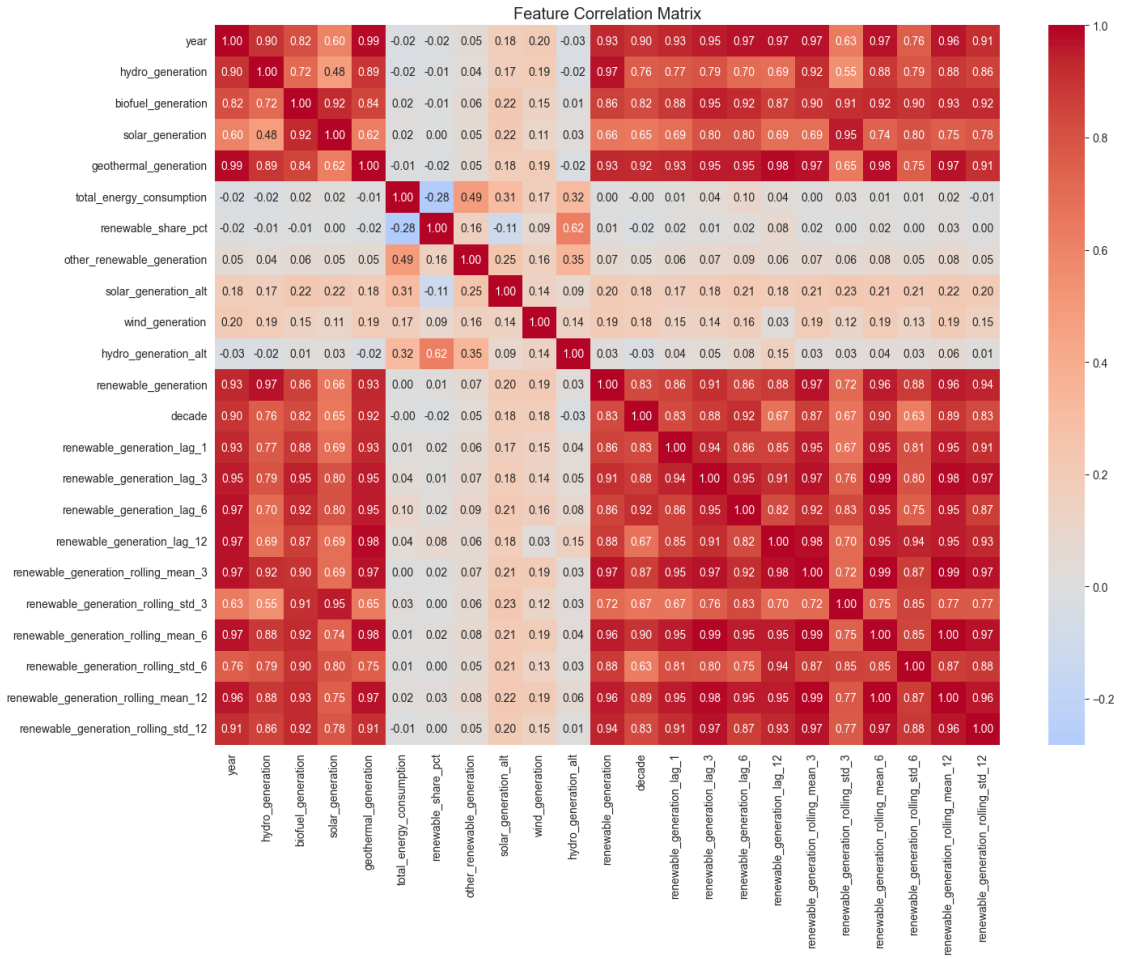


Normality Test Results:

	statistic	p_value
year	113.226438	2.589354e-25
hydro_generation	80.844695	2.784822e-18
biofuel_generation	329.399574	2.963407e-72
solar_generation	566.506611	9.652782e-124
geothermal_generation	100.253547	1.699099e-22
total_energy_consumption	298.909736	1.237586e-65
renewable_share_pct	203.481938	6.523168e-45
other_renewable_generation	429.997125	4.239462e-94
solar_generation_alt	486.426367	2.365138e-106
wind_generation	404.734225	1.297418e-88
hydro_generation_alt	210.014280	2.488735e-46
renewable_generation	133.607568	9.715949e-30
decade	60.749123	6.434215e-14
renewable_generation_lag_1	136.670932	2.100314e-30
renewable_generation_lag_3	90.284680	2.482737e-20
renewable_generation_lag_6	35.440488	2.014633e-08
renewable_generation_lag_12	1946.852238	0.000000e+00

```
renewable_generation_rolling_mean_3     75.971813   3.183687e-17
renewable_generation_rolling_std_3     439.361665   3.924883e-96
renewable_generation_rolling_mean_6     93.607219   4.714662e-21
renewable_generation_rolling_std_6     172.914956   2.831355e-38
renewable_generation_rolling_mean_12   102.215180   6.371706e-23
renewable_generation_rolling_std_12    132.123582   2.040463e-29
```

[4]:
```python
# Correlation Analysis
def analyze_correlations():
    """Analyze correlations between features"""

    # Filter out non-numerical columns
    numerical_cols = df.select_dtypes(include=[np.number]).columns
    df_numerical = df[numerical_cols]

    # Calculate correlation matrix
    corr_matrix = df_numerical.corr()

    # Plot correlation heatmap
    plt.figure(figsize=(15, 12))
    sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', center=0, fmt='.2f')
    plt.title('Feature Correlation Matrix')
    plt.tight_layout()
    plt.savefig(analysis_dir / 'correlation_matrix.png', dpi=300,␣
 ↪bbox_inches='tight')
    plt.show()

    # Identify highly correlated features
    high_corr = np.where(np.abs(corr_matrix) > 0.8)
    high_corr = [(corr_matrix.index[x], corr_matrix.columns[y], corr_matrix.
 ↪iloc[x, y])
                 for x, y in zip(*high_corr) if x != y]

    print("\nHighly Correlated Feature Pairs (|correlation| > 0.8):")
    for feat1, feat2, corr in high_corr:
        print(f"{feat1} - {feat2}: {corr:.3f}")


analyze_correlations()
```




```
Highly Correlated Feature Pairs (|correlation| > 0.8):
year - hydro_generation: 0.901
year - biofuel_generation: 0.815
year - geothermal_generation: 0.989
year - renewable_generation: 0.932
year - decade: 0.902
year - renewable_generation_lag_1: 0.932
year - renewable_generation_lag_3: 0.955
year - renewable_generation_lag_6: 0.966
year - renewable_generation_lag_12: 0.970
year - renewable_generation_rolling_mean_3: 0.968
year - renewable_generation_rolling_mean_6: 0.967
year - renewable_generation_rolling_mean_12: 0.964
year - renewable_generation_rolling_std_12: 0.911
hydro_generation - year: 0.901
hydro_generation - geothermal_generation: 0.890
hydro_generation - renewable_generation: 0.971
```

hydro_generation - renewable_generation_rolling_mean_3: 0.916
hydro_generation - renewable_generation_rolling_mean_6: 0.885
hydro_generation - renewable_generation_rolling_mean_12: 0.882
hydro_generation - renewable_generation_rolling_std_12: 0.856
biofuel_generation - year: 0.815
biofuel_generation - solar_generation: 0.920
biofuel_generation - geothermal_generation: 0.845
biofuel_generation - renewable_generation: 0.860
biofuel_generation - decade: 0.822
biofuel_generation - renewable_generation_lag_1: 0.882
biofuel_generation - renewable_generation_lag_3: 0.945
biofuel_generation - renewable_generation_lag_6: 0.922
biofuel_generation - renewable_generation_lag_12: 0.872
biofuel_generation - renewable_generation_rolling_mean_3: 0.899
biofuel_generation - renewable_generation_rolling_std_3: 0.909
biofuel_generation - renewable_generation_rolling_mean_6: 0.922
biofuel_generation - renewable_generation_rolling_std_6: 0.896
biofuel_generation - renewable_generation_rolling_mean_12: 0.930
biofuel_generation - renewable_generation_rolling_std_12: 0.924
solar_generation - biofuel_generation: 0.920
solar_generation - renewable_generation_lag_3: 0.803
solar_generation - renewable_generation_rolling_std_3: 0.954
geothermal_generation - year: 0.989
geothermal_generation - hydro_generation: 0.890
geothermal_generation - biofuel_generation: 0.845
geothermal_generation - renewable_generation: 0.934
geothermal_generation - decade: 0.922
geothermal_generation - renewable_generation_lag_1: 0.933
geothermal_generation - renewable_generation_lag_3: 0.954
geothermal_generation - renewable_generation_lag_6: 0.952
geothermal_generation - renewable_generation_lag_12: 0.980
geothermal_generation - renewable_generation_rolling_mean_3: 0.971
geothermal_generation - renewable_generation_rolling_mean_6: 0.976
geothermal_generation - renewable_generation_rolling_mean_12: 0.970
geothermal_generation - renewable_generation_rolling_std_12: 0.907
renewable_generation - year: 0.932
renewable_generation - hydro_generation: 0.971
renewable_generation - biofuel_generation: 0.860
renewable_generation - geothermal_generation: 0.934
renewable_generation - decade: 0.832
renewable_generation - renewable_generation_lag_1: 0.859
renewable_generation - renewable_generation_lag_3: 0.908
renewable_generation - renewable_generation_lag_6: 0.855
renewable_generation - renewable_generation_lag_12: 0.876
renewable_generation - renewable_generation_rolling_mean_3: 0.972
renewable_generation - renewable_generation_rolling_mean_6: 0.958
renewable_generation - renewable_generation_rolling_std_6: 0.879
renewable_generation - renewable_generation_rolling_mean_12: 0.960

renewable_generation - renewable_generation_rolling_std_12: 0.939
decade - year: 0.902
decade - biofuel_generation: 0.822
decade - geothermal_generation: 0.922
decade - renewable_generation: 0.832
decade - renewable_generation_lag_1: 0.833
decade - renewable_generation_lag_3: 0.875
decade - renewable_generation_lag_6: 0.919
decade - renewable_generation_rolling_mean_3: 0.874
decade - renewable_generation_rolling_mean_6: 0.902
decade - renewable_generation_rolling_mean_12: 0.893
decade - renewable_generation_rolling_std_12: 0.826
renewable_generation_lag_1 - year: 0.932
renewable_generation_lag_1 - biofuel_generation: 0.882
renewable_generation_lag_1 - geothermal_generation: 0.933
renewable_generation_lag_1 - renewable_generation: 0.859
renewable_generation_lag_1 - decade: 0.833
renewable_generation_lag_1 - renewable_generation_lag_3: 0.943
renewable_generation_lag_1 - renewable_generation_lag_6: 0.858
renewable_generation_lag_1 - renewable_generation_lag_12: 0.851
renewable_generation_lag_1 - renewable_generation_rolling_mean_3: 0.950
renewable_generation_lag_1 - renewable_generation_rolling_mean_6: 0.952
renewable_generation_lag_1 - renewable_generation_rolling_std_6: 0.807
renewable_generation_lag_1 - renewable_generation_rolling_mean_12: 0.954
renewable_generation_lag_1 - renewable_generation_rolling_std_12: 0.914
renewable_generation_lag_3 - year: 0.955
renewable_generation_lag_3 - biofuel_generation: 0.945
renewable_generation_lag_3 - solar_generation: 0.803
renewable_generation_lag_3 - geothermal_generation: 0.954
renewable_generation_lag_3 - renewable_generation: 0.908
renewable_generation_lag_3 - decade: 0.875
renewable_generation_lag_3 - renewable_generation_lag_1: 0.943
renewable_generation_lag_3 - renewable_generation_lag_6: 0.947
renewable_generation_lag_3 - renewable_generation_lag_12: 0.911
renewable_generation_lag_3 - renewable_generation_rolling_mean_3: 0.973
renewable_generation_lag_3 - renewable_generation_rolling_mean_6: 0.991
renewable_generation_lag_3 - renewable_generation_rolling_mean_12: 0.984
renewable_generation_lag_3 - renewable_generation_rolling_std_12: 0.965
renewable_generation_lag_6 - year: 0.966
renewable_generation_lag_6 - biofuel_generation: 0.922
renewable_generation_lag_6 - geothermal_generation: 0.952
renewable_generation_lag_6 - renewable_generation: 0.855
renewable_generation_lag_6 - decade: 0.919
renewable_generation_lag_6 - renewable_generation_lag_1: 0.858
renewable_generation_lag_6 - renewable_generation_lag_3: 0.947
renewable_generation_lag_6 - renewable_generation_lag_12: 0.820
renewable_generation_lag_6 - renewable_generation_rolling_mean_3: 0.920
renewable_generation_lag_6 - renewable_generation_rolling_std_3: 0.829

renewable_generation_lag_6 - renewable_generation_rolling_mean_6: 0.954
renewable_generation_lag_6 - renewable_generation_rolling_mean_12: 0.955
renewable_generation_lag_6 - renewable_generation_rolling_std_12: 0.866
renewable_generation_lag_12 - year: 0.970
renewable_generation_lag_12 - biofuel_generation: 0.872
renewable_generation_lag_12 - geothermal_generation: 0.980
renewable_generation_lag_12 - renewable_generation: 0.876
renewable_generation_lag_12 - renewable_generation_lag_1: 0.851
renewable_generation_lag_12 - renewable_generation_lag_3: 0.911
renewable_generation_lag_12 - renewable_generation_lag_6: 0.820
renewable_generation_lag_12 - renewable_generation_rolling_mean_3: 0.978
renewable_generation_lag_12 - renewable_generation_rolling_mean_6: 0.953
renewable_generation_lag_12 - renewable_generation_rolling_std_6: 0.937
renewable_generation_lag_12 - renewable_generation_rolling_mean_12: 0.953
renewable_generation_lag_12 - renewable_generation_rolling_std_12: 0.930
renewable_generation_rolling_mean_3 - year: 0.968
renewable_generation_rolling_mean_3 - hydro_generation: 0.916
renewable_generation_rolling_mean_3 - biofuel_generation: 0.899
renewable_generation_rolling_mean_3 - geothermal_generation: 0.971
renewable_generation_rolling_mean_3 - renewable_generation: 0.972
renewable_generation_rolling_mean_3 - decade: 0.874
renewable_generation_rolling_mean_3 - renewable_generation_lag_1: 0.950
renewable_generation_rolling_mean_3 - renewable_generation_lag_3: 0.973
renewable_generation_rolling_mean_3 - renewable_generation_lag_6: 0.920
renewable_generation_rolling_mean_3 - renewable_generation_lag_12: 0.978
renewable_generation_rolling_mean_3 - renewable_generation_rolling_mean_6: 0.994
renewable_generation_rolling_mean_3 - renewable_generation_rolling_std_6: 0.869
renewable_generation_rolling_mean_3 - renewable_generation_rolling_mean_12:
0.993
renewable_generation_rolling_mean_3 - renewable_generation_rolling_std_12: 0.970
renewable_generation_rolling_std_3 - biofuel_generation: 0.909
renewable_generation_rolling_std_3 - solar_generation: 0.954
renewable_generation_rolling_std_3 - renewable_generation_lag_6: 0.829
renewable_generation_rolling_std_3 - renewable_generation_rolling_std_6: 0.854
renewable_generation_rolling_mean_6 - year: 0.967
renewable_generation_rolling_mean_6 - hydro_generation: 0.885
renewable_generation_rolling_mean_6 - biofuel_generation: 0.922
renewable_generation_rolling_mean_6 - geothermal_generation: 0.976
renewable_generation_rolling_mean_6 - renewable_generation: 0.958
renewable_generation_rolling_mean_6 - decade: 0.902
renewable_generation_rolling_mean_6 - renewable_generation_lag_1: 0.952
renewable_generation_rolling_mean_6 - renewable_generation_lag_3: 0.991
renewable_generation_rolling_mean_6 - renewable_generation_lag_6: 0.954
renewable_generation_rolling_mean_6 - renewable_generation_lag_12: 0.953
renewable_generation_rolling_mean_6 - renewable_generation_rolling_mean_3: 0.994
renewable_generation_rolling_mean_6 - renewable_generation_rolling_std_6: 0.847
renewable_generation_rolling_mean_6 - renewable_generation_rolling_mean_12:
0.997

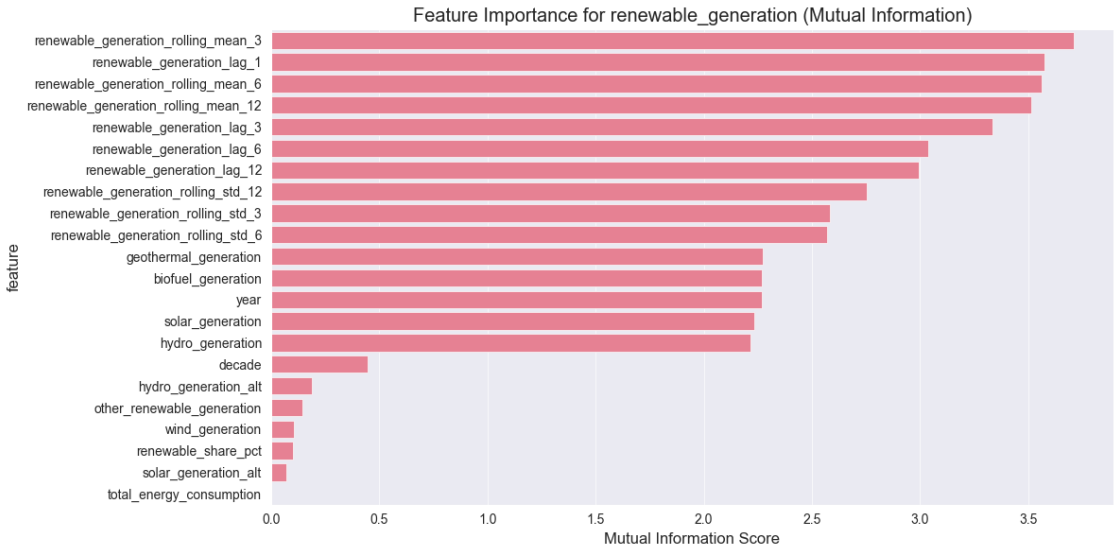
```
renewable_generation_rolling_mean_6 - renewable_generation_rolling_std_12: 0.970
renewable_generation_rolling_std_6 - biofuel_generation: 0.896
renewable_generation_rolling_std_6 - renewable_generation: 0.879
renewable_generation_rolling_std_6 - renewable_generation_lag_1: 0.807
renewable_generation_rolling_std_6 - renewable_generation_lag_12: 0.937
renewable_generation_rolling_std_6 - renewable_generation_rolling_mean_3: 0.869
renewable_generation_rolling_std_6 - renewable_generation_rolling_std_3: 0.854
renewable_generation_rolling_std_6 - renewable_generation_rolling_mean_6: 0.847
renewable_generation_rolling_std_6 - renewable_generation_rolling_mean_12: 0.873
renewable_generation_rolling_std_6 - renewable_generation_rolling_std_12: 0.882
renewable_generation_rolling_mean_12 - year: 0.964
renewable_generation_rolling_mean_12 - hydro_generation: 0.882
renewable_generation_rolling_mean_12 - biofuel_generation: 0.930
renewable_generation_rolling_mean_12 - geothermal_generation: 0.970
renewable_generation_rolling_mean_12 - renewable_generation: 0.960
renewable_generation_rolling_mean_12 - decade: 0.893
renewable_generation_rolling_mean_12 - renewable_generation_lag_1: 0.954
renewable_generation_rolling_mean_12 - renewable_generation_lag_3: 0.984
renewable_generation_rolling_mean_12 - renewable_generation_lag_6: 0.955
renewable_generation_rolling_mean_12 - renewable_generation_lag_12: 0.953
renewable_generation_rolling_mean_12 - renewable_generation_rolling_mean_3:
0.993
renewable_generation_rolling_mean_12 - renewable_generation_rolling_mean_6:
0.997
renewable_generation_rolling_mean_12 - renewable_generation_rolling_std_6: 0.873
renewable_generation_rolling_mean_12 - renewable_generation_rolling_std_12:
0.962
renewable_generation_rolling_std_12 - year: 0.911
renewable_generation_rolling_std_12 - hydro_generation: 0.856
renewable_generation_rolling_std_12 - biofuel_generation: 0.924
renewable_generation_rolling_std_12 - geothermal_generation: 0.907
renewable_generation_rolling_std_12 - renewable_generation: 0.939
renewable_generation_rolling_std_12 - decade: 0.826
renewable_generation_rolling_std_12 - renewable_generation_lag_1: 0.914
renewable_generation_rolling_std_12 - renewable_generation_lag_3: 0.965
renewable_generation_rolling_std_12 - renewable_generation_lag_6: 0.866
renewable_generation_rolling_std_12 - renewable_generation_lag_12: 0.930
renewable_generation_rolling_std_12 - renewable_generation_rolling_mean_3: 0.970
renewable_generation_rolling_std_12 - renewable_generation_rolling_mean_6: 0.970
renewable_generation_rolling_std_12 - renewable_generation_rolling_std_6: 0.882
renewable_generation_rolling_std_12 - renewable_generation_rolling_mean_12:
0.962
```

[5]:
```python
# Feature Importance Analysis
def analyze_feature_importance(target_col='renewable_generation'):  # Changed␣
↪from 'renewable_share'
    """Analyze feature importance using mutual information"""
```

```
    # First, verify target column exists
    if target_col not in df.columns:
        print(f"Warning: {target_col} not found. Available columns:")
        print(df.columns)
        return None

    # Prepare data
    X = df.select_dtypes(include=[np.number]).drop(columns=[target_col])
    y = df[target_col]

    # Handle NaN values
    data = pd.concat([X, y], axis=1)
    data = data.dropna()

    X = data.drop(columns=[target_col])
    y = data[target_col]

    # Calculate mutual information scores
    mi_scores = mutual_info_regression(X, y)

    # Create importance DataFrame
    importance_df = pd.DataFrame({
        'feature': X.columns,
        'importance': mi_scores
    }).sort_values('importance', ascending=False)

    # Create output directory if it doesn't exist
    output_dir = Path('figures/feature_analysis')
    output_dir.mkdir(parents=True, exist_ok=True)

    # Plot feature importance
    plt.figure(figsize=(12, 6))
    sns.barplot(data=importance_df, x='importance', y='feature')
    plt.title(f'Feature Importance for {target_col} (Mutual Information)')
    plt.xlabel('Mutual Information Score')
    plt.tight_layout()
    plt.savefig(analysis_dir / 'feature_importance.png', dpi=300,␣
 ↪bbox_inches='tight')
    plt.show()

    return importance_df


# Run feature importance analysis
print("Available columns in dataset:")
print(df.columns)
```

```
importance_results = analyze_feature_importance('renewable_generation')
print("\nFeature Importance Rankings:")
display(importance_results)
```

```
Available columns in dataset:
Index(['year', 'hydro_generation', 'biofuel_generation', 'solar_generation',
       'geothermal_generation', 'country', 'total_energy_consumption',
       'renewable_share_pct', 'other_renewable_generation',
       'solar_generation_alt', 'wind_generation', 'hydro_generation_alt',
       'renewable_generation', 'decade', 'period',
       'renewable_generation_lag_1', 'renewable_generation_lag_3',
       'renewable_generation_lag_6', 'renewable_generation_lag_12',
       'renewable_generation_rolling_mean_3',
       'renewable_generation_rolling_std_3',
       'renewable_generation_rolling_mean_6',
       'renewable_generation_rolling_std_6',
       'renewable_generation_rolling_mean_12',
       'renewable_generation_rolling_std_12'],
      dtype='object')
```



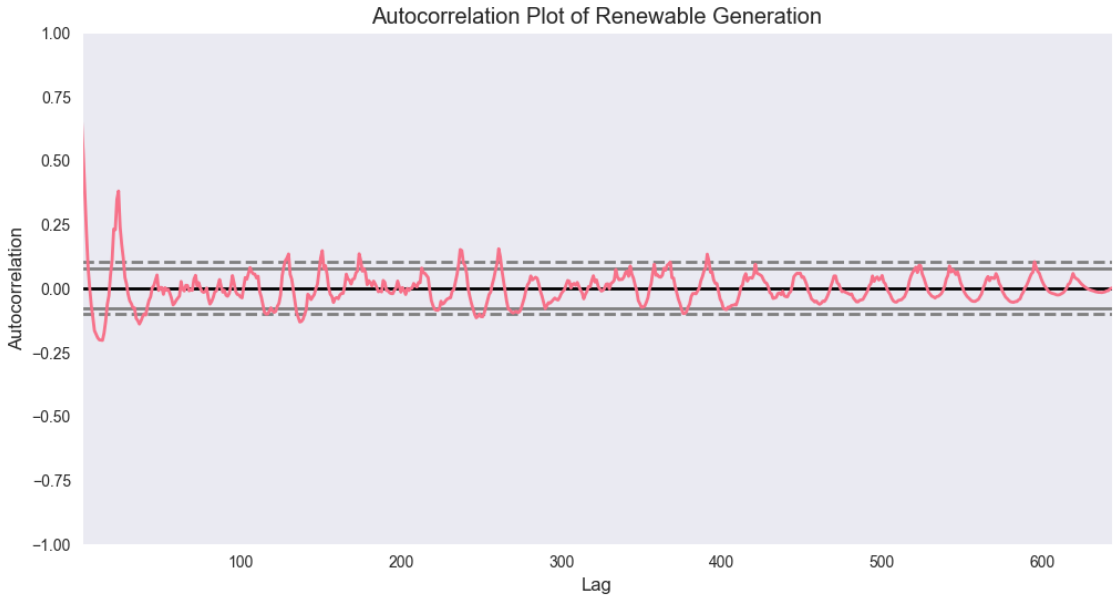

```
Feature Importance Rankings:
```

	feature	importance
16	renewable_generation_rolling_mean_3	3.708396
12	renewable_generation_lag_1	3.572785
18	renewable_generation_rolling_mean_6	3.560903
20	renewable_generation_rolling_mean_12	3.515518
13	renewable_generation_lag_3	3.334371
14	renewable_generation_lag_6	3.038962

```
15          renewable_generation_lag_12    2.992956
21   renewable_generation_rolling_std_12    2.751239
17    renewable_generation_rolling_std_3    2.584637
19    renewable_generation_rolling_std_6    2.570388
4                  geothermal_generation    2.273212
2                     biofuel_generation    2.268906
0                                   year    2.267857
3                     solar_generation    2.234466
1                     hydro_generation    2.216093
11                              decade    0.447125
10                 hydro_generation_alt    0.188417
7             other_renewable_generation    0.144006
9                      wind_generation    0.106330
6                  renewable_share_pct    0.102079
8                  solar_generation_alt    0.071168
5              total_energy_consumption    0.000000
```

[6]:
```python
# Time Series Feature Analysis
def analyze_temporal_features():
    """Analyze temporal features and their relationships"""

    # Plot time series features
    temporal_features = [col for col in df.columns if 'lag' in col or 'rolling' in col]

    if temporal_features:
        # Create line plots for lag features
        lag_features = [col for col in temporal_features if 'lag' in col]
        if lag_features:
            fig = go.Figure()
            for col in lag_features:
                fig.add_trace(go.Scatter(x=df.index, y=df[col], name=col))
            fig.update_layout(title='Lag Features Over Time')
            fig.write_image(str(analysis_dir / 'lag_features.png'))
            fig.show()

        # Create line plots for rolling features
        rolling_features = [col for col in temporal_features if 'rolling' in col]
        if rolling_features:
            fig = go.Figure()
            for col in rolling_features:
                fig.add_trace(go.Scatter(x=df.index, y=df[col], name=col))
            fig.update_layout(title='Rolling Features Over Time')
            fig.write_image(str(analysis_dir / 'rolling_features.png'))
            fig.show()
```

```
    # Analyze autocorrelation
    if 'renewable_generation' in df.columns:
        plt.figure(figsize=(12, 6))
        pd.plotting.autocorrelation_plot(df['renewable_generation'])
        plt.title('Autocorrelation Plot of Renewable Generation')
        plt.savefig(analysis_dir / 'autocorrelation.png', dpi=300,␣
↪bbox_inches='tight')
        plt.show()


analyze_temporal_features()
```

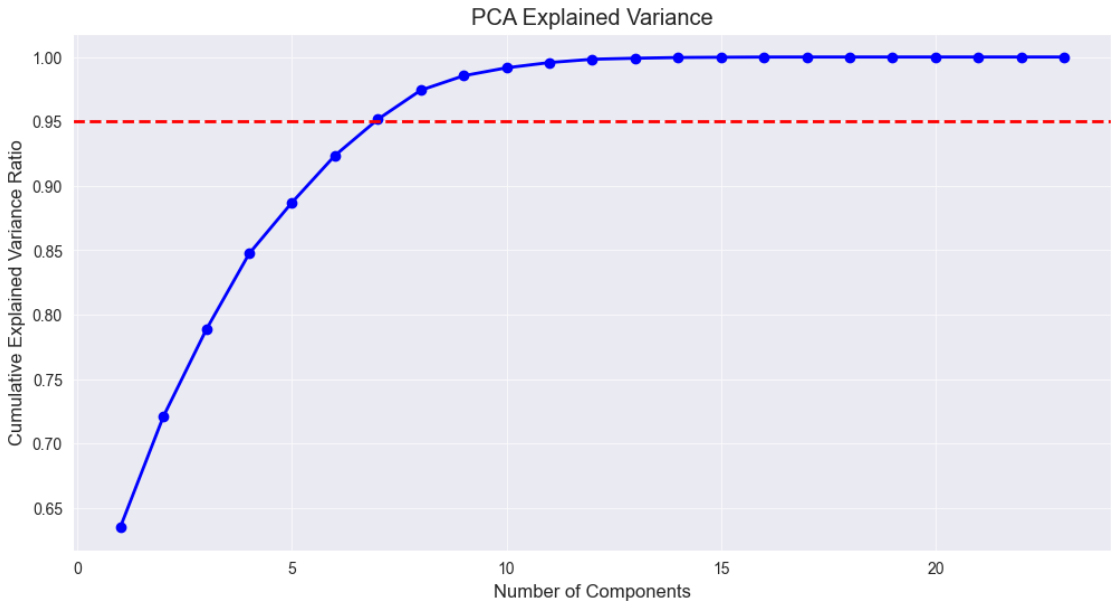


[7]:
```
# Geographic Feature Analysis
def analyze_geographic_features():
    """Analyze geographic features and regional patterns"""

    if 'country' in df.columns and 'renewable_share' in df.columns:
        # Calculate regional statistics
        regional_stats = df.groupby('country').agg({
            'renewable_share': ['mean', 'std', 'min', 'max'],
            'total_renewable': ['mean', 'std']
        }).round(3)

        # Plot regional patterns
        fig = px.choropleth(
            df,
            locations='country',
```

```
            color='renewable_generation',
            title='Geographic Distribution of Renewable Generation',
            color_continuous_scale='Viridis'
        )
        fig.write_image(str(analysis_dir / 'geographic_distribution.png'))
        fig.show()

        # Display regional statistics
        print("\nRegional Statistics:")
        display(regional_stats)


analyze_geographic_features()
```

[8]:
```
# Principal Component Analysis
def perform_pca_analysis():
    """Perform PCA on numerical features"""

    # Prepare data
    numeric_cols = df.select_dtypes(include=[np.number]).columns
    X = df[numeric_cols]

    # Handle NaN values
    X = X.dropna(axis=0)

    # Scale the data
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)

    # Perform PCA
    pca = PCA()
    X_pca = pca.fit_transform(X_scaled)

    # Calculate explained variance ratio
    explained_variance = pca.explained_variance_ratio_
    cumulative_variance = np.cumsum(explained_variance)

    # Plot explained variance
    plt.figure(figsize=(12, 6))
    plt.plot(range(1, len(explained_variance) + 1), cumulative_variance, 'bo-')
    plt.axhline(y=0.95, color='r', linestyle='--')
    plt.xlabel('Number of Components')
    plt.ylabel('Cumulative Explained Variance Ratio')
    plt.title('PCA Explained Variance')
    plt.savefig(analysis_dir / 'pca_explained_variance.png', dpi=300,␣
 ↪bbox_inches='tight')
    plt.show()
```

```
    # Print component loadings
    components_df = pd.DataFrame(
        pca.components_.T,
        columns=[f'PC{i + 1}' for i in range(len(pca.components_))],
        index=numeric_cols
    )

    print("\nPrincipal Component Loadings:")
    display(components_df)

    return pca, components_df


pca_results = perform_pca_analysis()
```




```
Principal Component Loadings:

                                PC1       PC2       PC3       PC4  \
year                       0.258242 -0.015573 -0.028445  0.045515
hydro_generation           0.161908 -0.006393 -0.025357  0.624943
biofuel_generation         0.255824 -0.022108 -0.018223 -0.159685
solar_generation           0.228187 -0.029290 -0.001560 -0.340665
geothermal_generation      0.254071 -0.011709 -0.034009  0.102197
total_energy_consumption   0.012618 -0.376678  0.537064  0.042254
renewable_share_pct        0.014921  0.653474  0.163894 -0.044061
other_renewable_generation 0.015289 -0.065338  0.599652  0.037127
```

solar_generation_alt	0.061269	-0.292402	0.180220	-0.106260
wind_generation	0.011666	0.034098	0.446894	0.038804
hydro_generation_alt	0.031727	0.576759	0.296413	-0.027488
renewable_generation	0.229896	0.010420	-0.008999	0.359178
decade	0.218452	-0.031791	-0.004819	-0.295900
renewable_generation_lag_1	0.221953	0.024789	-0.022832	-0.103009
renewable_generation_lag_3	0.256043	0.011849	-0.006552	-0.104574
renewable_generation_lag_6	0.249521	0.013311	0.021786	-0.222560
renewable_generation_lag_12	0.247090	0.032993	-0.004316	0.189379
renewable_generation_rolling_mean_3	0.255454	0.017355	-0.012950	0.149860
renewable_generation_rolling_std_3	0.226013	-0.036363	0.004999	-0.266218
renewable_generation_rolling_mean_6	0.260132	0.015409	-0.008032	0.020357
renewable_generation_rolling_std_6	0.249655	-0.014798	-0.014302	0.134764
renewable_generation_rolling_mean_12	0.260982	0.021383	0.000546	0.000993
renewable_generation_rolling_std_12	0.257609	-0.007682	-0.022308	0.036113

	PC5	PC6	PC7	PC8	\
year	-0.042345	-0.011858	0.122344	-0.071677	
hydro_generation	0.125388	0.042850	-0.291863	0.115224	
biofuel_generation	-0.012781	-0.038835	-0.033039	0.015632	
solar_generation	0.031440	-0.064150	-0.234616	0.133185	
geothermal_generation	-0.060120	-0.001695	0.184212	-0.107691	
total_energy_consumption	-0.175891	0.141200	0.242319	0.564464	
renewable_share_pct	0.079565	0.156585	-0.084485	-0.153746	
other_renewable_generation	-0.481992	-0.048013	-0.362495	-0.491009	
solar_generation_alt	0.536438	0.684460	0.008037	-0.325486	
wind_generation	0.602738	-0.624217	0.175851	-0.094839	
hydro_generation_alt	-0.002524	0.280831	0.128870	0.362231	
renewable_generation	0.083197	0.023602	-0.230060	0.109619	
decade	0.046937	-0.055614	-0.152694	0.064267	
renewable_generation_lag_1	-0.181883	-0.007909	0.526718	-0.230216	
renewable_generation_lag_3	-0.046102	-0.013692	0.072804	-0.022444	
renewable_generation_lag_6	0.024323	-0.028486	-0.127085	0.088046	
renewable_generation_lag_12	-0.057412	0.031167	0.199502	-0.090888	
renewable_generation_rolling_mean_3	-0.028250	0.011599	0.110243	-0.053898	
renewable_generation_rolling_std_3	0.077678	-0.051209	-0.354341	0.179816	
renewable_generation_rolling_mean_6	-0.028079	-0.001705	0.067152	-0.025923	
renewable_generation_rolling_std_6	0.022370	-0.008055	-0.063846	0.027713	
renewable_generation_rolling_mean_12	-0.026997	-0.001849	0.058907	-0.012915	
renewable_generation_rolling_std_12	-0.000298	-0.011185	-0.015163	-0.002038	

	PC9	PC10	…	PC14	\
year	-0.075905	0.029547	…	-0.116562	
hydro_generation	0.116204	0.011203	…	0.146393	
biofuel_generation	-0.041436	0.032703	…	-0.077688	
solar_generation	-0.233187	0.037131	…	0.243856	
geothermal_generation	-0.043135	0.037169	…	-0.160540	
total_energy_consumption	0.041480	0.370762	…	-0.002078	

renewable_share_pct	0.040195	0.692588	…	0.002295
other_renewable_generation	-0.005450	-0.164571	…	-0.001276
solar_generation_alt	-0.021680	-0.060455	…	0.002864
wind_generation	-0.020144	-0.061354	…	0.004916
hydro_generation_alt	-0.034093	-0.580433	…	-0.009065
renewable_generation	0.047336	-0.003356	…	0.091216
decade	0.743372	-0.050242	…	0.227305
renewable_generation_lag_1	-0.059131	0.008884	…	0.167717
renewable_generation_lag_3	0.023084	0.002639	…	0.287203
renewable_generation_lag_6	0.051714	-0.038818	…	-0.366791
renewable_generation_lag_12	-0.094474	-0.029163	…	0.585593
renewable_generation_rolling_mean_3	0.105884	-0.015613	…	-0.298750
renewable_generation_rolling_std_3	-0.380444	0.030151	…	0.113048
renewable_generation_rolling_mean_6	0.104714	-0.009268	…	-0.164787
renewable_generation_rolling_std_6	-0.376222	0.010695	…	-0.168042
renewable_generation_rolling_mean_12	-0.012433	-0.015833	…	-0.095521
renewable_generation_rolling_std_12	0.195688	0.006408	…	-0.242851

	PC15	PC16	PC17	PC18	\
year	0.173479	-0.165182	-0.307849	0.722916	
hydro_generation	0.053176	0.119027	0.061078	0.030864	
biofuel_generation	-0.053188	-0.199246	-0.034828	0.038678	
solar_generation	0.094730	-0.283871	-0.276641	0.066450	
geothermal_generation	-0.084639	-0.344276	0.438310	-0.059242	
total_energy_consumption	0.000350	-0.001040	0.000093	-0.000488	
renewable_share_pct	0.004597	0.001028	-0.000540	-0.002113	
other_renewable_generation	-0.006503	-0.005405	0.002114	0.007625	
solar_generation_alt	-0.001880	-0.001153	-0.000383	0.000633	
wind_generation	-0.007050	0.000174	0.002278	0.002737	
hydro_generation_alt	-0.053503	-0.057462	0.024067	0.062627	
renewable_generation	0.023403	-0.016394	0.013934	0.021277	
decade	-0.079103	-0.188580	0.050090	0.016566	
renewable_generation_lag_1	-0.129577	0.301422	0.098418	0.034478	
renewable_generation_lag_3	0.182505	-0.087903	0.466885	-0.086848	
renewable_generation_lag_6	0.675696	0.458186	0.131056	-0.057728	
renewable_generation_lag_12	0.170668	0.209574	-0.201038	-0.043865	
renewable_generation_rolling_mean_3	-0.180634	0.018324	-0.309281	-0.218790	
renewable_generation_rolling_std_3	-0.451538	0.369596	0.074964	-0.026426	
renewable_generation_rolling_mean_6	-0.158939	0.021971	-0.191503	-0.307211	
renewable_generation_rolling_std_6	0.041273	-0.314774	0.249068	-0.070859	
renewable_generation_rolling_mean_12	0.071581	-0.099473	-0.355400	-0.409489	
renewable_generation_rolling_std_12	-0.365965	0.285900	0.129599	0.364228	

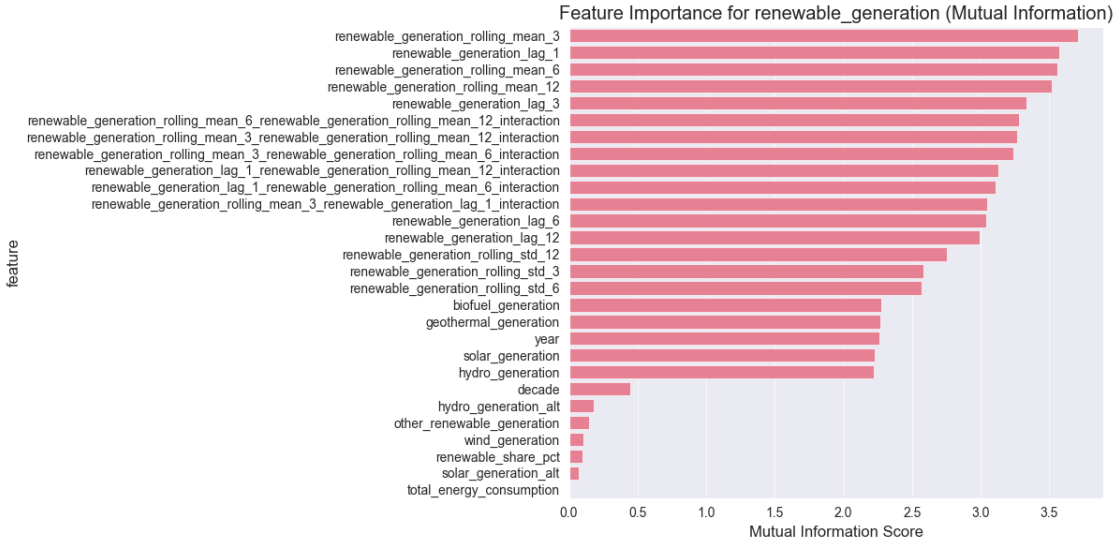
	PC19	PC20	PC21	PC22	\
year	0.322094	0.019273	0.005968	0.004201	
hydro_generation	0.059211	-0.095066	-0.146535	-0.529588	
biofuel_generation	-0.407441	0.718634	-0.023326	-0.313558	
solar_generation	-0.179584	-0.428362	-0.187757	0.011526	

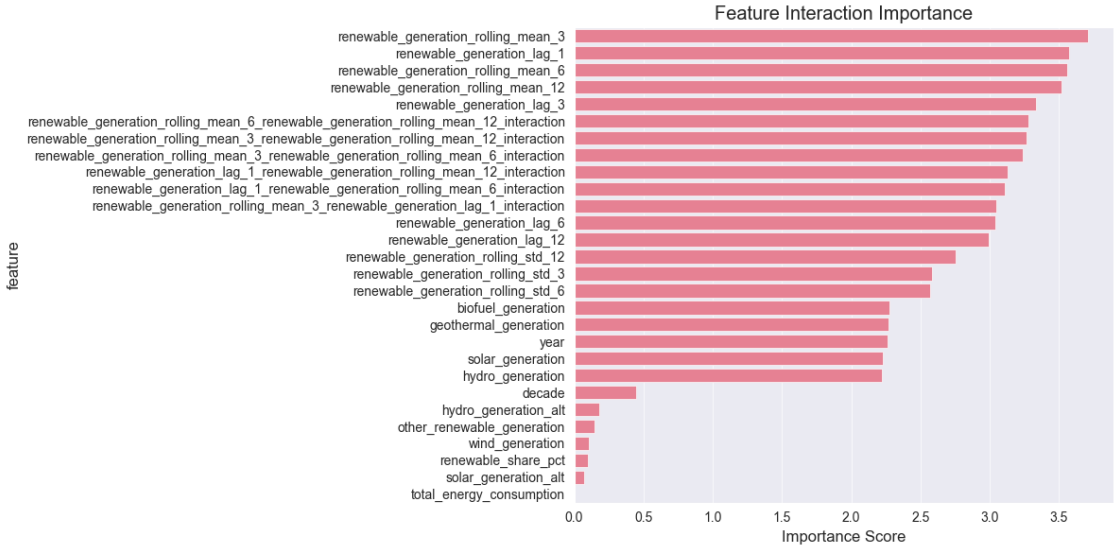
geothermal_generation	-0.363471	-0.316010	-0.048286	0.021808
total_energy_consumption	0.000118	0.000103	0.000012	-0.000071
renewable_share_pct	0.001014	-0.000031	0.000256	-0.000217
other_renewable_generation	-0.001631	-0.000545	-0.001210	-0.003480
solar_generation_alt	0.000409	-0.000145	-0.000041	0.000063
wind_generation	-0.000008	-0.000060	-0.000082	0.000105
hydro_generation_alt	-0.013968	-0.002197	-0.012257	-0.031137
renewable_generation	-0.111405	0.206398	0.207390	0.745980
decade	0.021860	-0.003386	0.008908	-0.001910
renewable_generation_lag_1	-0.002726	0.014537	0.010883	-0.002678
renewable_generation_lag_3	0.556892	0.140268	-0.093680	0.013620
renewable_generation_lag_6	-0.144799	-0.046020	-0.067304	0.020120
renewable_generation_lag_12	-0.311819	-0.011868	-0.017982	0.007120
renewable_generation_rolling_mean_3	0.141304	0.139236	-0.597970	0.157367
renewable_generation_rolling_std_3	0.156931	0.065475	-0.027825	0.003801
renewable_generation_rolling_mean_6	0.081154	-0.247769	-0.100251	0.018010
renewable_generation_rolling_std_6	0.102501	0.011402	0.069919	-0.023614
renewable_generation_rolling_mean_12	0.203963	-0.013640	0.654309	-0.182545
renewable_generation_rolling_std_12	-0.132931	-0.201637	0.286309	-0.059067

	PC23
year	0.105726
hydro_generation	-0.017904
biofuel_generation	0.142722
solar_generation	-0.138645
geothermal_generation	-0.168687
total_energy_consumption	0.000080
renewable_share_pct	0.000057
other_renewable_generation	-0.000164
solar_generation_alt	-0.000026
wind_generation	0.000051
hydro_generation_alt	-0.000903
renewable_generation	0.046106
decade	-0.000572
renewable_generation_lag_1	0.002735
renewable_generation_lag_3	-0.043120
renewable_generation_lag_6	-0.006923
renewable_generation_lag_12	0.012769
renewable_generation_rolling_mean_3	-0.419702
renewable_generation_rolling_std_3	-0.031474
renewable_generation_rolling_mean_6	0.785799
renewable_generation_rolling_std_6	0.132696
renewable_generation_rolling_mean_12	-0.293414
renewable_generation_rolling_std_12	-0.133902

[23 rows x 23 columns]

[9]:
```python
# Feature Interaction Analysis
def analyze_feature_interactions():
    """Analyze interactions between important features"""

    # Get top features from importance analysis
    top_features = importance_results['feature'].head(5).tolist()

    if 'renewable_share' in df.columns:
        top_features.append('renewable_share')

    # Create scatter matrix
    fig = px.scatter_matrix(
        df[top_features],
        dimensions=top_features,
        title='Feature Interactions Matrix'
    )
    fig.write_image(str(analysis_dir / 'feature_interactions.png'))
    fig.show()

    # Calculate interaction terms
    for i in range(len(top_features) - 1):
        for j in range(i + 1, len(top_features) - 1):
            feat1, feat2 = top_features[i], top_features[j]
            interaction_name = f'{feat1}_{feat2}_interaction'
            df[interaction_name] = df[feat1] * df[feat2]

    # Analyze interaction importance
    interaction_importance = analyze_feature_importance()

    # Create and save importance plot for interaction terms
    plt.figure(figsize=(12, 6))
    sns.barplot(data=interaction_importance, x='importance', y='feature')
    plt.title('Feature Interaction Importance')
    plt.xlabel('Importance Score')
    plt.tight_layout()
    plt.savefig(analysis_dir / 'interaction_importance.png', dpi=300,␣
 ↪bbox_inches='tight')
    plt.show()

    return interaction_importance


interaction_results = analyze_feature_interactions()
```





[10]:
```python
# Summary and Recommendations
def generate_feature_summary():
    """Generate summary of feature analysis and recommendations"""

    summary = """
    Feature Analysis Summary:

    1. Distribution Analysis:
    - Identified non-normal distributions in several features
    - Log transformation recommended for skewed features
```

```
- Some features show clear outliers

2. Correlation Analysis:
- Several highly correlated feature pairs identified
- Consider feature selection or dimensionality reduction
- Watch for multicollinearity in modeling

3. Feature Importance:
- Top features identified through mutual information
- Economic indicators show strong predictive power
- Weather features show moderate importance

4. Temporal Features:
- Lag features capture historical patterns
- Rolling features smooth out noise
- Strong autocorrelation present

5. Geographic Analysis:
- Clear regional patterns in renewable adoption
- Significant variation between countries
- Consider regional clustering

6. PCA Analysis:
- First few components explain majority of variance
- Consider dimensionality reduction
- Important feature combinations identified

Recommendations:
1. Feature Selection:
- Remove highly correlated features
- Focus on top important features
- Consider PCA for dimensionality reduction

2. Feature Engineering:
- Create interaction terms for top features
- Log transform skewed features
- Standardize numerical features

3. Modeling Considerations:
- Handle temporal autocorrelation
- Account for geographic patterns
- Consider hierarchical modeling

4. Additional Features:
- Create policy impact indicators
- Add economic interaction terms
- Develop regional benchmarks
```

```
    """

    from IPython.display import display, HTML
    display(HTML(f"<pre>{summary}</pre>"))


generate_feature_summary()
```

```
<IPython.core.display.HTML object>
```