In [10]:
```
#Question 3

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import itertools
import statsmodels.api as sm
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [11]:
```
#Load the dataset
df = pd.read_csv("C:\\Users\\OkechPC\\Downloads\\FE-GWP1_model_selection_1.csv")
print("Data shape:", df.shape)
print(df.head(), "\n")
```

```
Data shape: (100, 6)
          Y        X1        X2        X3        X4        X5
0  3.388410  0.017954 -0.800583 -0.352454  2.187210  1.014887
1  0.287191  0.083057 -0.597947 -0.357639 -1.630284  0.221841
2  3.989645 -0.923437 -1.386575  1.180202  0.632606 -1.576638
3 -2.959602 -0.313775  2.955133 -1.798692 -2.117621  0.159291
4  0.529773  0.388996  1.019611  0.472062  0.590497  0.877048
```

In [12]:
```
#All subsets and information criterion

y = df["Y"]
predictors = ["X1", "X2", "X3", "X4", "X5"]

results = []
for k in range(1, len(predictors)+1):
    for subset in itertools.combinations(predictors, k):
        X = sm.add_constant(df[list(subset)])
        model = sm.OLS(y, X).fit()
        results.append({
            "subset": subset,
            "k": k,
            "adj_R2": model.rsquared_adj,
            "AIC": model.aic,
            "BIC": model.bic
        })

res_df = pd.DataFrame(results)
print("Top 5 models by Adjusted R²")
print(res_df.sort_values("adj_R2", ascending=False).head(), "\n")
print("Top 5 models by AIC")
print(res_df.sort_values("AIC").head(), "\n")
print("Top 5 models by BIC")
print(res_df.sort_values("BIC").head(), "\n")
```

```
Top 5 models by Adjusted R²
                  subset  k    adj_R2         AIC         BIC
29      (X2, X3, X4, X5)  4  0.633974  260.616684  273.642535
30  (X1, X2, X3, X4, X5)  5  0.630170  262.592528  278.223549
21          (X2, X3, X4)  3  0.616639  264.291054  274.711735
25      (X1, X2, X3, X4)  4  0.612991  266.191097  279.216948
23          (X2, X4, X5)  3  0.494796  291.889597  302.310277

Top 5 models by AIC
                  subset  k    adj_R2         AIC         BIC
29      (X2, X3, X4, X5)  4  0.633974  260.616684  273.642535
30  (X1, X2, X3, X4, X5)  5  0.630170  262.592528  278.223549
21          (X2, X3, X4)  3  0.616639  264.291054  274.711735
25      (X1, X2, X3, X4)  4  0.612991  266.191097  279.216948
23          (X2, X4, X5)  3  0.494796  291.889597  302.310277

Top 5 models by BIC
                  subset  k    adj_R2         AIC         BIC
29      (X2, X3, X4, X5)  4  0.633974  260.616684  273.642535
21          (X2, X3, X4)  3  0.616639  264.291054  274.711735
30  (X1, X2, X3, X4, X5)  5  0.630170  262.592528  278.223549
25      (X1, X2, X3, X4)  4  0.612991  266.191097  279.216948
23          (X2, X4, X5)  3  0.494796  291.889597  302.310277
```

In [13]:
```python
#Choose the best model (highest adj R²)

best = res_df.sort_values("adj_R2", ascending=False).iloc[0]
best_predictors = list(best["subset"])
print(f"Best model by Adjusted R² uses predictors: {best_predictors}\n")
```

```
Best model by Adjusted R² uses predictors: ['X2', 'X3', 'X4', 'X5']
```

In [14]:
```python
#Fitting Best model

X_best = sm.add_constant(df[best_predictors])
best_model = sm.OLS(y, X_best).fit()
print(best_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.649
Model:                            OLS   Adj. R-squared:                  0.634
Method:                 Least Squares   F-statistic:                     43.87
Date:                Sun, 14 Sep 2025   Prob (F-statistic):           8.29e-21
Time:                        10:38:51   Log-Likelihood:                -125.31
No. Observations:                 100   AIC:                             260.6
Df Residuals:                      95   BIC:                             273.6
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.1893      0.089     13.333      0.000       1.012       1.366
X2            -0.5861      0.091     -6.440      0.000      -0.767      -0.405
X3             0.5592      0.091      6.124      0.000       0.378       0.740
X4             0.7105      0.082      8.672      0.000       0.548       0.873
X5            -0.1966      0.083     -2.355      0.021      -0.362      -0.031
==============================================================================
Omnibus:                        4.462   Durbin-Watson:                   1.974
Prob(Omnibus):                  0.107   Jarque-Bera (JB):                3.920
Skew:                           0.473   Prob(JB):                        0.141
Kurtosis:                       3.215   Cond. No.                         1.39
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

In [15]:
```
#Multicollinearity diagnostics: VIF

vif_data = pd.DataFrame({
    "Variable": X_best.columns,
    "VIF": [variance_inflation_factor(X_best.values, i)
            for i in range(X_best.shape[1])]
})
print("\nVariance Inflation Factors")
print(vif_data, "\n")
```

```
Variance Inflation Factors
  Variable       VIF
0    const  1.053255
1       X2  1.004719
2       X3  1.041876
3       X4  1.020243
4       X5  1.052515
```

In [16]:
```
# Residuals

resid = best_model.resid
print("Residual mean (≈0):", resid.mean())
print("Residual std:", resid.std(ddof=1))
print("Skewness:", resid.skew())
print("Kurtosis:", resid.kurtosis())
```

```
Residual mean (≈0): 6.461498003318411e-16
Residual std: 0.8514368505026784
Skewness: 0.48012577300088544
Kurtosis: 0.2889013868807311
```