Insights and Conclusions from the ACF Plots

Residual Patterns: The top-left plot shows significant autocorrelation for the first few lags, suggesting that the residuals from the Passenger\_Count model are not completely white noise and there might be some remaining structure that the model hasn't captured.

Other Residuals: The other three plots (top-right, bottom-left, bottom-right) indicate that there are no significant autocorrelations in the residuals for those variables, implying that those residuals behave like white noise.

Model Fit: The presence of significant autocorrelation in the top-left plot suggests that the Passenger\_Count model might be underfitting and missing some patterns in the data.

Stability: The residuals for Station\_Pair\_Encoded and other variables being close to zero and within the confidence bounds indicate stability and potentially good fit for these variables.

Need for Refinement: The overall patterns suggest that while the model captures most relationships well, there might be room for improvement, particularly for the Passenger\_Count variable.

Key Takeaways from the VAR Model

Strong Autoregressive Relationship: The high coefficient for L1.Passenger\_Count (0.862506) indicates a strong autoregressive relationship for Passenger\_Count, meaning past values strongly influence current values.

Interdependence: The significant coefficient for L1.Station\_Pair\_Encoded in the Passenger\_Count equation (0.137494) shows some interdependence between Passenger\_Count and Station\_Pair\_Encoded.

Model Performance: The VAR model, with its FPE, AIC, BIC, and HQIC values, suggests a reasonably good fit but highlights areas (like Passenger\_Count) that could benefit from additional modeling techniques or variables.

Precision and Accuracy: The high t-statistics and low p-values for most coefficients suggest the model parameters are estimated precisely, with the effects of L1.Passenger\_Count and L1.Station\_Pair\_Encoded being statistically significant.

Residual Correlation: The low correlation between residuals of Passenger\_Count and Station\_Pair\_Encoded indicates that the model adequately separates the influences of these variables.

Explanation of the Graph

Top-Left Plot: This ACF plot for the residuals of Passenger\_Count shows significant autocorrelations for the first few lags, suggesting that the model's residuals are not purely random.

Top-Right Plot: This plot indicates that there is no significant autocorrelation for the residuals of another variable (possibly Station\_Pair\_Encoded), as all values are within the confidence bounds.

Bottom-Left Plot: Similarly, this ACF plot shows no significant autocorrelation, indicating that the residuals are white noise.

Bottom-Right Plot: Again, no significant autocorrelation is observed, reinforcing the idea that the residuals for these variables do not exhibit systematic patterns.

Overall: The combined plots suggest that while the model performs well for most variables, there may be some issues with the residuals of Passenger\_Count that need further investigation.

What's Happening in This Notebook?

Reading Data: Imagine you have a huge book that keeps track of how many people got on and off buses at different stops every hour for a few years. This notebook opens that book and reads all the information.

Preparing the Data: The data in the book has information like the time, the bus stops, and the number of people. The notebook makes sure all the dates and times are in the right format so it can understand them better.

Encoding Stops: Bus stops have names like "Stop A" and "Stop B". The notebook changes these names into numbers because it’s easier for the computer to work with numbers.

Model Training: The notebook uses a special program called a "VAR model" to learn from the past data. This program looks at how many people were at the stops at different times and tries to understand the patterns.

Making Predictions: After learning from the past, the notebook tries to guess how many people will be at the bus stops in the next few hours. It's like trying to predict how many friends will come to your birthday party based on how many came in the past years.

Steps Explained Simply:

Open the book and read all the pages to see how many people used the buses at different times.

Make sure the dates and times are easy to understand.

Change bus stop names into numbers.

Teach the computer to look at the past data and understand the patterns.

Ask the computer to guess how many people will use the buses in the next few hours based on what it has learned.

What Does the VAR Model Predict?

The Vector Autoregression (VAR) model is used to predict future values of multiple time series data. In this notebook, the VAR model is used to predict:

Passenger\_Count: The number of passengers using the buses at different times.

Station\_Pair\_Encoded: The encoded value of the bus stops pairs.

How Good Are the Predictions? Can They Be Trusted?

The quality and trustworthiness of the predictions depend on several factors:

Model Evaluation Metrics: Common metrics include Mean Squared Error (MSE), Root Mean Squared Error (RMSE), or Mean Absolute Error (MAE). These can help to understand how close the predicted values are to the actual values.

Residual Analysis: Checking the residuals (differences between actual and predicted values) to see if they are randomly distributed. If they show patterns, the model might be missing something.

Stationarity: The time series data needs to be stationary (its properties do not change over time) for VAR models to work well. If the data is not stationary, transformations like differencing might be needed.

Out-of-Sample Testing: Checking how well the model performs on data it hasn't seen before.

In the notebook, it looks like only the model summary is printed, but to trust the predictions, additional steps should be performed:

Calculate evaluation metrics.

Analyze residuals.

Perform out-of-sample testing.

How Can It Be Improved?

If the predictions are not good enough, several steps can be taken to improve the model:

Data Preprocessing: Ensure the data is stationary. If not, apply transformations like differencing.

Feature Engineering: Add more relevant features that could help the model understand the patterns better.

Hyperparameter Tuning: Adjust the lag length of the VAR model to find the optimal number of previous time steps to use.

Model Selection: Try other time series models like ARIMA, SARIMA, or machine learning models like Random Forests or Neural Networks.

Cross-Validation: Use techniques like cross-validation to ensure the model is not overfitting and can generalize well to new data.

Inputs and Outputs for the VAR Model

Inputs:

Multiple time series data. In this case:

Passenger\_Count

Station\_Pair\_Encoded

Outputs:

Predicted values of the input time series data for the specified number of steps ahead.

How Does It Work Internally?

Lagged Values: The VAR model uses the past values of all the time series variables to predict their future values. For example, to predict Passenger\_Count and Station\_Pair\_Encoded at time t, it uses the values of both variables at previous times t-1, t-2, ..., t-p, where p is the lag order.

Linear Relationships: The model assumes a linear relationship between the current values of the time series and their past values. The model can be represented as:

𝑌

𝑡

=

𝑐

+

𝐴

1

𝑌

𝑡

−

1

+

𝐴

2

𝑌

𝑡

−

2

+

.

.

.

+

𝐴

𝑝

𝑌

𝑡

−

𝑝

+

𝜖

𝑡

Y

t

​

=c+A

1

​

Y

t−1

​

+A

2

​

Y

t−2

​

+...+A

p

​

Y

t−p

​

+ϵ

t

​

where:

𝑌

𝑡

Y

t

​

is a vector of the variables (here Passenger\_Count and Station\_Pair\_Encoded).

𝑐

c is a vector of constants.

𝐴

𝑖

A

i

​

are coefficient matrices.

𝜖

𝑡

ϵ

t

​

is a vector of error terms.

Estimation: The coefficients are estimated using Ordinary Least Squares (OLS) for each equation.

Forecasting: Once the model is fitted, it can use the estimated coefficients and the past values of the variables to predict future values.

Summary

The VAR model in the notebook predicts the number of passengers and the encoded bus stop pairs based on past data. To assess how good the predictions are, you should look at evaluation metrics and residuals. If the predictions need improvement, consider better data preprocessing, feature engineering, and trying other models or hyperparameters. The VAR model works by finding linear relationships between past values of multiple time series to predict their future values.

2. ACF and PACF Plots of Residuals

The Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots can help in understanding the autocorrelation in residuals, which indicates if the model has adequately captured the temporal dependencies.

python

Copy code

from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf

fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(14, 8))

for i, col in enumerate(residuals.columns):

plot\_acf(residuals[col], lags=50, ax=axes[i, 0], title=f'ACF of {col} Residuals')

plot\_pacf(residuals[col], lags=50, ax=axes[i, 1], title=f'PACF of {col} Residuals')

plt.tight\_layout()

plt.show()

3. Impulse Response Functions (IRF)

Impulse Response Functions show how the variables in the VAR model respond to a shock in one of the variables over time.

python

Copy code

irf = results.irf(10)

irf.plot(orth=False)

plt.show()

4. Forecast Error Variance Decomposition (FEVD)

FEVD shows the proportion of the forecast error variance of each variable that can be attributed to shocks in the other variables.

python

Copy code

fevd = results.fevd(10)

fevd.plot()

plt.show()

5. Lag Order Selection Criteria

If not already done, it’s useful to visualize criteria for selecting the optimal lag order, such as AIC, BIC, HQIC.

python

Copy code

lag\_order\_results = model.select\_order(maxlags=15)

print(lag\_order\_results.summary())

# Plot the criteria

criteria = lag\_order\_results.aic, lag\_order\_results.bic, lag\_order\_results.hqic

plt.plot(criteria[0], label='AIC')

plt.plot(criteria[1], label='BIC')

plt.plot(criteria[2], label='HQIC')

plt.legend()

plt.xlabel('Lag Order')

plt.ylabel('Criterion Value')

plt.title('Lag Order Selection Criteria')

plt.show()

6. Forecast Plots

Visualize the forecasts and compare them to the actual data.

python

Copy code

# Make a prediction for the next 5 time steps

forecast = results.forecast(data.values[-results.k\_ar:], steps=5)

forecast\_df = pd.DataFrame(forecast, index=pd.date\_range(start=data.index[-1], periods=6, closed='right'), columns=data.columns)

# Plot the forecasted vs actual values

fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(10, 8))

for i, col in enumerate(data.columns):

axes[i].plot(data[col].iloc[-100:], label='Actual')

axes[i].plot(forecast\_df[col], label='Forecast')

axes[i].set\_title(f'Forecast vs Actual for {col}')

axes[i].legend()

plt.tight\_layout()

plt.show()

Insights from the Residuals Graph

Graph Explanation

Top Graph: Residuals of Passenger\_Count

Overview: The top graph displays the residuals of the Passenger\_Count variable over time. Residuals are the differences between the observed values and the values predicted by the VAR model.

Time Range: The x-axis spans from 2018 to 2024, indicating the timeline over which the data was analyzed.

Magnitude of Residuals: The y-axis shows the residual values, which range approximately from -300 to +400.

Bottom Graph: Residuals of Station\_Pair\_Encoded

Overview: The bottom graph shows the residuals of the Station\_Pair\_Encoded variable over the same time range.

Time Range: Like the top graph, the x-axis covers the period from 2018 to 2024.

Magnitude of Residuals: The y-axis here ranges from -200 to +200.

Key Insights

Residual Variability Over Time (Passenger\_Count):

The residuals for Passenger\_Count show significant variability over time. The early years (2018-2019) exhibit more pronounced fluctuations, suggesting potential outliers or seasonality effects that the model didn't capture well.

Over time, especially post-2020, the magnitude of residuals appears to stabilize somewhat but still shows considerable variance, indicating persistent challenges in the model's predictive accuracy for this variable.

Model Fit Quality (Passenger\_Count):

The wide range of residual values (both positive and negative) suggests that the model might not be capturing all the dynamics influencing passenger counts effectively. This could point to either an overfitting or underfitting issue or the presence of external factors not included in the model.

Station\_Pair\_Encoded Residual Patterns:

The residuals for Station\_Pair\_Encoded display a distinct pattern where the residuals remain relatively close to zero, with less variability compared to Passenger\_Count. This suggests that the model predicts the Station\_Pair\_Encoded variable with higher accuracy or that the encoding has a relatively stable pattern over time.

The funnel-like shape indicates that the residuals converge to zero over time, which might suggest that as more data becomes available, the model's predictions for this variable improve.

Potential Model Improvements:

Given the higher residuals for Passenger\_Count, it may be beneficial to incorporate additional variables or lag terms in the VAR model to better capture the underlying trends and seasonal patterns. Exploring different transformations of the data or including external factors such as holidays or events might improve the model fit.

Diagnosis of Station\_Pair\_Encoded Residuals:

The relatively narrow range and consistent pattern of residuals for Station\_Pair\_Encoded indicate that the encoding process was effective, but there might be room to explore if additional features or different encoding techniques could further enhance the predictive power for this variable.

Conclusion

The residual plots provide valuable insights into the performance of the VAR model. While the model captures some aspects of the data, there are noticeable discrepancies, especially with the Passenger\_Count variable. Addressing these through additional data features, model refinement, and exploring external influences could enhance the overall model accuracy and reliability. The residuals analysis is a crucial step in diagnosing model performance and guiding further improvements.

Explanation of Impulse Response Function (IRF)

The Impulse Response Function (IRF) is a crucial tool in analyzing the dynamic behavior of a VAR (Vector Autoregression) model. It measures the reaction of one variable in the system to a shock in another variable, illustrating how the impact of the shock evolves over time.

Graph Explanation and Insights

Top Left Graph: Passenger\_Count → Passenger\_Count

Explanation: This graph shows the response of Passenger\_Count to a shock in itself. The y-axis represents the magnitude of the response, while the x-axis shows the time periods (lags).

Insight 1: The response starts at a value of 1, indicating an immediate effect of the shock.

Insight 2: Over time, the impact decays gradually towards zero, suggesting that the shock's influence diminishes and eventually becomes negligible after about 10 periods.

Top Right Graph: Station\_Pair\_Encoded → Passenger\_Count

Explanation: This graph depicts the response of Passenger\_Count to a shock in Station\_Pair\_Encoded.

Insight 3: The response starts at zero and increases steadily over time. This indicates that a shock in Station\_Pair\_Encoded has a growing positive influence on Passenger\_Count, potentially reflecting a delayed but cumulative effect.

Bottom Left Graph: Passenger\_Count → Station\_Pair\_Encoded

Explanation: This graph shows the response of Station\_Pair\_Encoded to a shock in Passenger\_Count.

Insight 4: The response is very small, almost negligible, indicating that shocks in Passenger\_Count have a minimal and gradual effect on Station\_Pair\_Encoded. The confidence intervals (dotted lines) suggest that this effect is statistically insignificant.

Bottom Right Graph: Station\_Pair\_Encoded → Station\_Pair\_Encoded

Explanation: This graph illustrates the response of Station\_Pair\_Encoded to a shock in itself.

Insight 5: The response remains constant at 1 over time, indicating a persistent and steady effect. This suggests that Station\_Pair\_Encoded has a strong self-influence, maintaining the impact of a shock without decay over the observed periods.

Detailed Insights

Passenger\_Count Self-Influence:

Decay Over Time: The Passenger\_Count variable's self-response shows a classic decay pattern, highlighting that the influence of a shock to itself diminishes over time. This indicates that while a change in passenger count has an immediate effect, the system gradually returns to equilibrium.

Delayed Cumulative Impact:

Growing Influence: The Station\_Pair\_Encoded to Passenger\_Count graph shows a steadily increasing response, suggesting that station pair changes have a delayed cumulative effect on passenger count. This could be due to route adjustments or gradual passenger behavior changes in response to new station pairings.

Minimal Cross-Variable Impact:

Negligible Effect: The minimal response of Station\_Pair\_Encoded to Passenger\_Count shocks suggests that passenger count changes do not significantly influence station pair dynamics, pointing to a one-way dependency where station configurations affect passenger numbers more than the reverse.

Persistent Self-Influence:

Steady Response: The constant response in the Station\_Pair\_Encoded to itself indicates that changes in station pair configurations are persistent. This could reflect the stable nature of station infrastructure and routes once established.

Conclusion

The IRF plots provide a comprehensive view of the dynamic interactions between Passenger\_Count and Station\_Pair\_Encoded. The primary insights suggest that passenger counts are influenced by station configurations over time, but the reverse influence is minimal. Additionally, passenger count shocks tend to diminish over time, while station pair configurations maintain a steady impact. These findings can inform strategies for optimizing station layouts and predicting passenger flows based on infrastructure changes.

VAR Order Selection Explanation

What is VAR Order Selection?

VAR (Vector Autoregression) order selection involves determining the optimal lag length for the VAR model. The optimal lag length is crucial as it affects the model's predictive power and accuracy.

Criteria for VAR Order Selection:

AIC (Akaike Information Criterion)

BIC (Bayesian Information Criterion)

FPE (Final Prediction Error)

HQIC (Hannan-Quinn Information Criterion)

These criteria help to balance the model fit and complexity. Lower values generally indicate a better model.

Summary of Order Selection:

The provided output shows the values of these criteria for different lag orders (0 to 15). The goal is to identify the lag order with the minimum values for these criteria.

From the summary:

Minimum Values:

AIC: 2.995 at lag 15

BIC: 2.995 at lag 15

FPE: 19.98 at lag 15

HQIC: 2.995 at lag 15

The asterisks (\*) indicate the minimum values for each criterion, suggesting that the optimal lag length for this model is 15.

Why the Plot is Empty

The issue with the plot being empty could be due to the way the criteria are being plotted. The criteria list contains tuples of AIC, BIC, and HQIC values, but the code should handle them properly to ensure they are plotted correctly.

Explanation of the VAR Order Selection Summary

AIC (Akaike Information Criterion): A measure of the model's fit quality, taking into account the number of parameters. Lower values indicate a better fit.

BIC (Bayesian Information Criterion): Similar to AIC, but includes a stronger penalty for models with more parameters. Lower values indicate a better fit.

HQIC (Hannan-Quinn Information Criterion): Another criterion similar to AIC and BIC, with a different penalty term. Lower values indicate a better fit.

In your summary, the criteria values decrease as the lag order increases. The lowest values for AIC, BIC, and HQIC are at lag order 15, which suggests that this is the optimal lag order for your VAR model.

If you still want to extract these values programmatically, you may need to revisit how the lag\_order\_results object is structured and make sure it's not storing individual values but rather arrays/lists.

Insights from the Lag Order Selection Criteria Graph

Sharp Decline at the Start:

Insight: There is a significant drop in the criterion values from lag order 0 to lag order 1. This suggests that including at least one lag in the model dramatically improves its performance.

Explanation: Initially, the model has no lagged terms, so adding the first lag brings substantial new information, improving the model's fit considerably.

Stabilization after Initial Drop:

Insight: After the sharp decline from lag order 0 to 1, the criterion values stabilize and show only marginal improvements with additional lags.

Explanation: Once the key lag(s) are included, the additional lags contribute less unique information, hence the criterion values stabilize.

Optimal Lag Order:

Insight: The lowest criterion values are observed at lag order 15 for AIC, BIC, and HQIC. This indicates that the optimal lag order for the VAR model, as per these criteria, is 15.

Explanation: The model with 15 lags balances the complexity and the fit to the data optimally according to the given criteria.

Consistency Across Criteria:

Insight: All three criteria (AIC, BIC, HQIC) show a similar trend, reinforcing the reliability of the optimal lag order determination.

Explanation: When multiple criteria agree, it strengthens the confidence in the selected model order being appropriate.

Understanding the Criteria Values:

Insight: The actual values of AIC, BIC, and HQIC are close to each other and tend to overlap after lag order 1, indicating that the penalty terms used in these criteria result in similar model evaluation metrics.

Explanation: AIC, BIC, and HQIC are all used to measure model fit while penalizing for the number of parameters. Their similar values post lag order 1 suggest the model fit is stable with minimal overfitting.

Story Behind the Data:

The graph represents the trade-off between model complexity and goodness of fit in a VAR model for different lag orders. Initially, adding more lag terms to the model greatly improves its performance, as seen by the sharp drop from lag order 0 to 1. However, as more lags are included, the improvements become marginal, indicating diminishing returns. This suggests that while the initial lags capture significant information from the data, additional lags add less incremental value.

The criteria (AIC, BIC, HQIC) help identify the optimal lag order by balancing fit quality against model complexity. The stabilization of criteria values after a certain point implies that the model has captured most of the relevant information, and further lags do not significantly enhance the model.

Ultimately, the lowest criteria values at lag order 15 indicate this as the optimal choice, as it provides the best fit with a manageable level of complexity. This helps in constructing a more accurate and efficient VAR model, which is crucial for effective forecasting and analysis.

By following these insights, one can ensure that the VAR model is neither underfitted nor overfitted, leading to more reliable and interpretable results.