Streaming Data Management and Time Series Analysis

Faezeh Azhir – 909890

2025.02.16

Time Series Forecasting Using ARIMA

1. Introduction

In this project, we have a hourly data which is time series data and has missing (period) observation which requires to predict based on the given periods. To overlook the objectives of the project we use ARIMA and SARIMA models prediction of the missing period in the data. First, we will use the cross-validation methods to make test and validation data, then we use the chosen models to get the best model among the class of models. Furthermore, we will have to choose the model for the prediction of the missing values while using the model performance.

2. Data Preparation and Analysis

The dataset was loaded and preprocessed to ensure correct datetime formatting and missing value handling. The main steps included:

- Converting the DateTime column to a proper format.
- Setting DateTime as the index for time-series analysis.
- Identifying and handling missing values.
- Splitting the dataset into training (80%) and validation (20%) sets to evaluate model performance.

A seasonality analysis was performed to detect daily patterns, confirming that the data had a strong 24-hour cycle.

3. Model Selection

To determine the best forecasting approach, we compared two models:

3.1 ARIMA Model

The ARIMA model requires three key parameters:

- p (AutoRegressive term) - Number of past observations used to predict the future.
- d (Differencing term) - Number of times the data is differenced to make it stationary.
- q (Moving Average term) - Number of past forecast errors included in the model.

Using Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots, we determined appropriate values for p, d, and q.

3.2 SARIMA Model

Since the data exhibited seasonality, we extended ARIMA to SARIMA, incorporating seasonal components:

- P (Seasonal AR term)
- D (Seasonal differencing)
- Q (Seasonal MA term)
- s (Seasonal period, set to 24 hours)

Multiple SARIMA models were tested, and the best parameters were selected based on Akaike Information Criterion (AIC), a measure of model quality.

4. Model Evaluation

The models were trained using the training dataset (80%) and evaluated on the validation dataset (20%). The accuracy of predictions was assessed using the following metrics:

- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- Mean Absolute Percentage Error (MAPE) (Adjusted to SMAPE due to division-by-zero issues)
- Akaike Information Criteria (AIC)
- Bayesian Information Criteria (BIC)

The table below displays the model's evaluation, such that we have to decide the model for the prediction in the basis of different model evaluation i.e. MSE, MAE, AIC and BIC. The lower value of the corresponding methods indicates the best model among the class of models.

Table-1: Models Evaluation

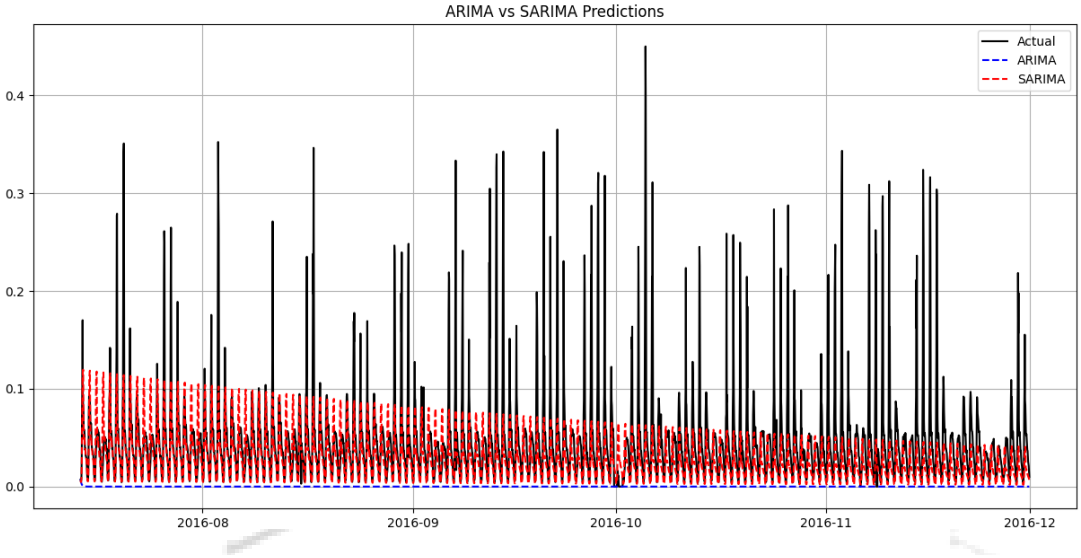
Model	MSE	RMSE	MAE	AIC	BIC
ARIMA	0.0047	0.0682	0.0467	-56771.66	-56734.14
SARIMA	0.002	0.0444	0.0239	-62233.63	-62181.09

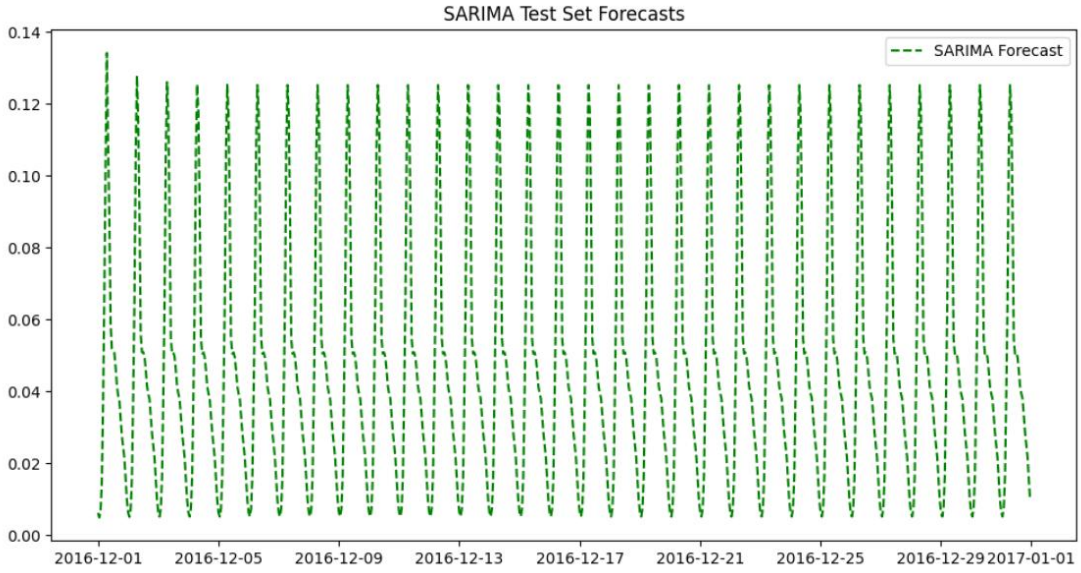
- SARIMA outperformed ARIMA, demonstrating better forecasting accuracy.
- ARIMA failed to capture seasonality, leading to higher error.

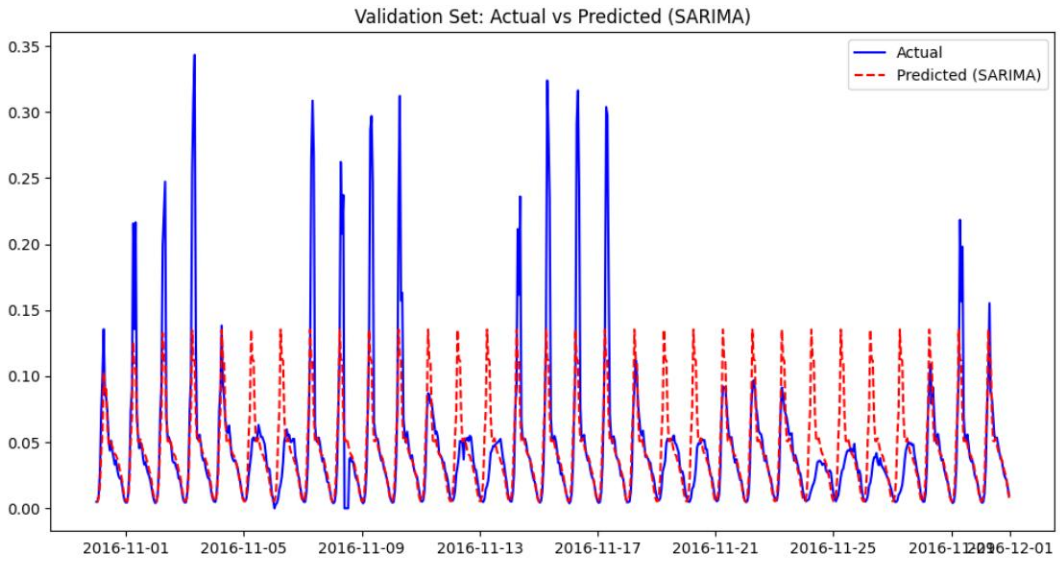
5. Forecasting Missing Values

After validating model performance:

1. The SARIMA model was retrained on the entire available dataset (including the validation period).
2. The missing period (test set) was forecasted using this final model.
3. The predicted values were stored in a CSV file for further evaluation.







Time Series Forecasting Using Unobserved Components Model (UCM)

1. Introduction

To achieve, I tested two variations of UCM, the objective was to determine which model provides more accurate forecasts and then use the best model for predicting missing values.

2. Data Preparation and Preprocessing

- The dataset was loaded and converted into a proper time-series format.
- The DateTime column was set as the index for easier analysis.
- Missing values were identified and separated for forecasting.
- The dataset was split into training (80%) and validation (20%) to evaluate model performance.

A seasonality analysis revealed a strong 24-hour pattern, indicating that the model should incorporate a seasonal component.

3. Model Selection: UCM Variants

I implemented and compared two types of UCM models:

3.1 Local Level Model

- Assumes that the time series follows a random walk with noise.
- Does not explicitly capture trend changes, making it suitable for stationary series.

3.2 Local Linear Trend Model

- Extends the Local Level model by adding a time-dependent trend component.
- Suitable for series where trends change over time rather than remaining constant.

Both models included a 24-hour seasonality component to account for daily fluctuations.

4. Model Evaluation and Comparison

To compare the models, they were trained on the training dataset (80%) and evaluated on the validation dataset (20%) using the following metrics:

- Mean Squared Error (MSE) – Lower values indicate better model performance.
- Root Mean Squared Error (RMSE) – Measures overall prediction error.
- Mean Absolute Error (MAE) – Evaluates average absolute differences.

Model Performance:

Table-2: Model Evaluation			
Model	**MSE**	**RMSE**	**MAE**
Local Level	0.001678	0.040966	0.018493
Local Linear Trend	0.001677	0.040954	0.018491

Key Observations: We have used another two models local linear trend and local level for the forecasting of missing observations. However, table-2 display the model evaluation of the selected models. Local Linear Trend had slightly lower error values, indicating a better fit. Both models performed similarly, but the Local Linear Trend model captured variations slightly better.

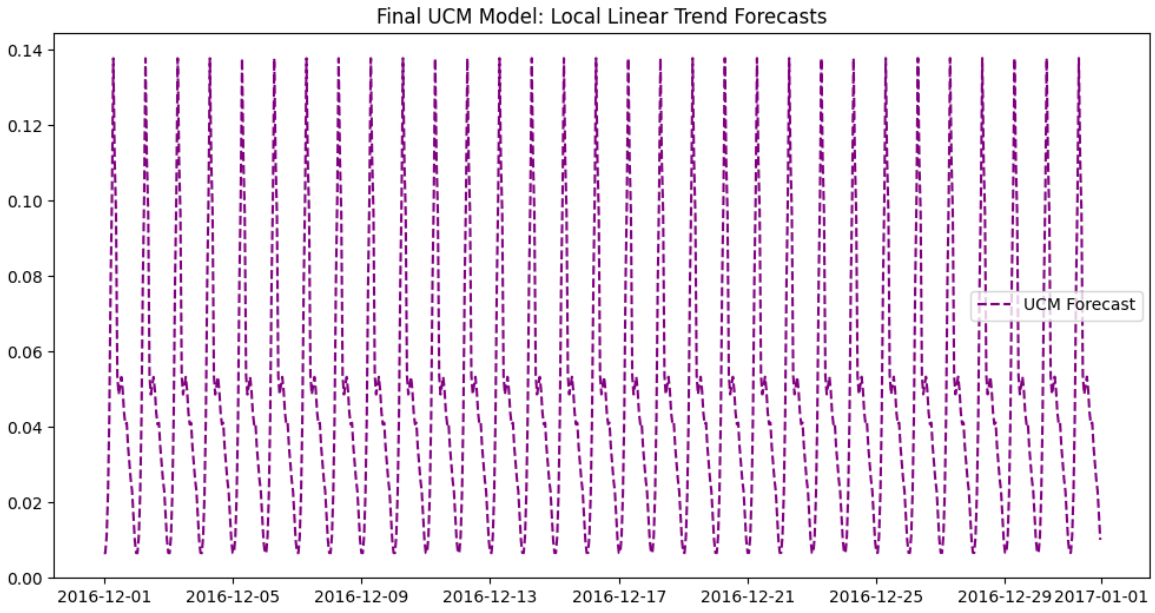
Best Model Selected: Local Linear Trend

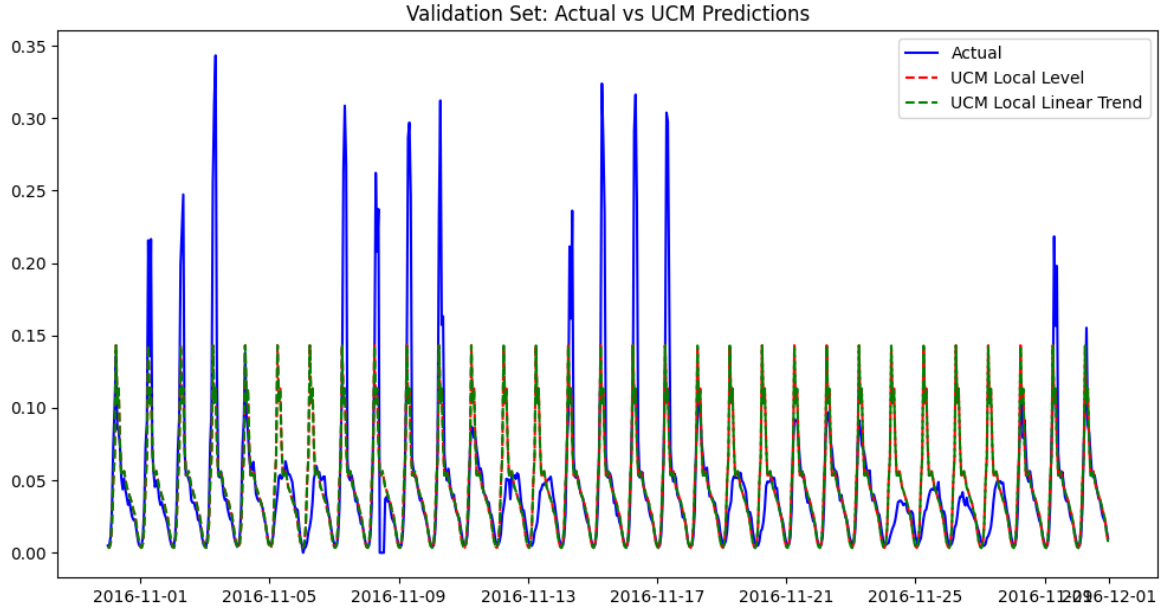
Since the Local Linear Trend Model had the lowest MSE, it was chosen for forecasting the missing values.

5. Forecasting Missing Values

Once the Local Linear Trend Model was identified as the best model, it was:

1. Retrained using the entire dataset (including validation data).
2. Used to forecast the missing values in the test set.
3. The predicted values were saved in a CSV file (UCM_Best_Model_Predictions.csv).


Final UCM Model: Local Linear Trend Forecasts
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
2016-12-01
2016-12-05
2016-12-09
2016-12-13
2016-12-17
2016-12-21
2016-12-25
2016-12-29
2017-01-01
UCM Forecast


Validation Set: Actual vs UCM Predictions
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
2016-11-01
2016-11-05
2016-11-09
2016-11-13
2016-11-17
2016-11-21
2016-11-25
2016-12-01
Actual
UCM Local Level
UCM Local Linear Trend

Time Series Forecasting Using Machine Learning Models

1. Introduction

Finally, we use different machine learning models for the prediction purpose. However, before using the model we used the cross-validation methods to classify the data in test and validation set for the best prediction of the data. After fitting the model we will evaluate the best model among the class of ML models for the prediction of the missing observations.

2. Data Preparation and Feature Engineering

The dataset was processed as follows:

- The DateTime column was formatted correctly and set as the index.
- Missing values were identified and separated for later forecasting.
- The dataset was split into training (80%) and validation (20%) for model evaluation.

Feature engineering was performed to enhance model performance:

- *Lag Features:* The last 24 hours of data were used as input features.
- *Time-based Features:* Extracted hour of the day and day of the week to capture periodic patterns.
- Missing values were handled using forward and backward filling to ensure consistency.

3. Model Selection and Training

Each of the three models was trained on the training dataset (80%) and validated on the remaining 20%. The following machine learning techniques were implemented:

3.1 XGBoost

- A gradient boosting algorithm optimized for efficiency and accuracy.
- Handles missing data internally and is effective in capturing complex patterns.

3.2 Gradient Boosting Regressor

- Similar to XGBoost but slower and less efficient in handling large datasets.
- Learns sequentially and tries to minimize residual errors.

3.3 Linear Regression

- A simple statistical model that assumes a linear relationship between variables.
- Used as a baseline model for comparison.

4. Model Evaluation and Comparison

To compare model performance, the following evaluation metrics were used:

- MSE (Mean Squared Error): Measures the average squared difference between actual and predicted values.
- RMSE (Root Mean Squared Error): The square root of MSE, indicating the model's prediction accuracy.
- MAE (Mean Absolute Error): Represents the absolute differences between predicted and actual values.

Performance Comparison:

Tabel-3: Models Evaluation

Model	MSE	RMSE	MAE
XGBoost	0.0002604	0.0161	0.0060
Gradient Boosting	0.0002755	0.0166	0.0072
Linear Regression	0.0005470	0.0233	0.0116

Key Observations: XGBoost had the lowest MSE, RMSE, and MAE, making it the most accurate model. Gradient Boosting performed well but was slightly worse than XGBoost. Linear Regression had the highest error, proving that a simple linear model is insufficient for time series forecasting.

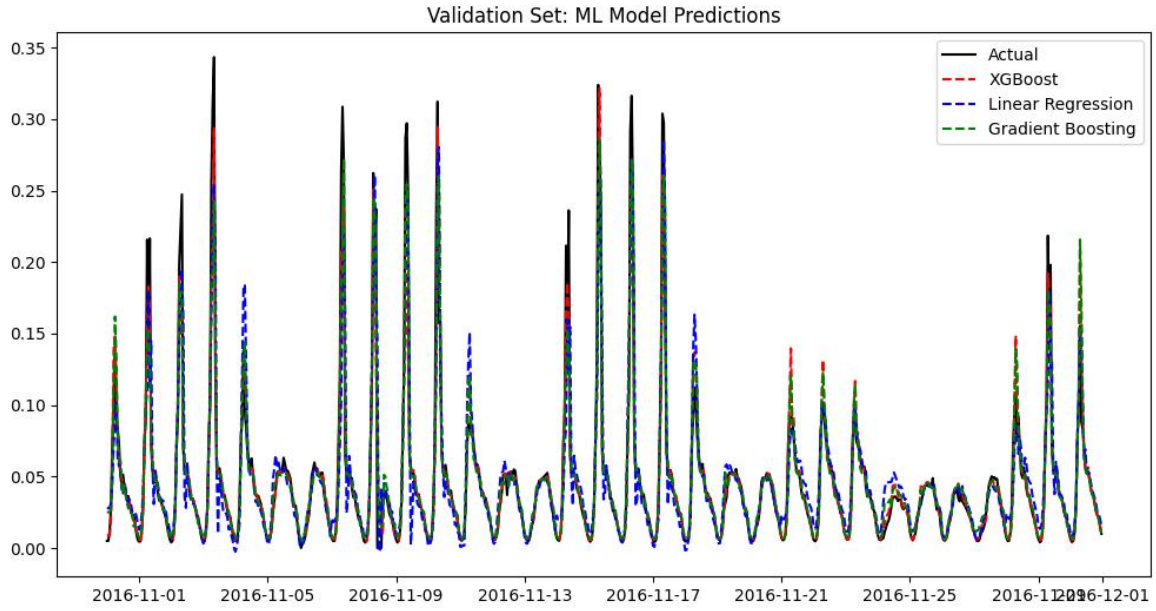
Best Model Selected: XGBoost

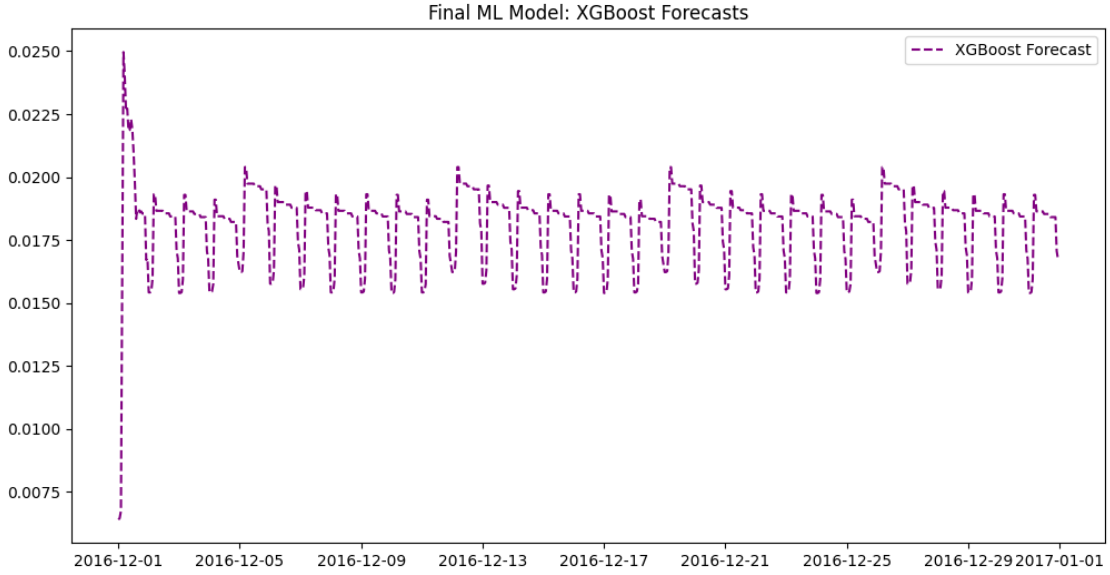
Since XGBoost had the lowest error values, it was chosen as the best model for forecasting the missing values.

5. Forecasting Missing Values

After selecting XGBoost as the best model:

1. The model was retrained on the full dataset (including validation data).
2. It was used to predict the missing values in the test set.
3. The final predictions were saved in a CSV file (ML_Best_Model_Predictions.csv).


Validation Set: ML Model Predictions
Actual
XGBoost
Linear Regression
Gradient Boosting
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
2016-11-01
2016-11-05
2016-11-09
2016-11-13
2016-11-17
2016-11-21
2016-11-25
2016-12-01


Final ML Model: XGBoost Forecasts
XGBoost Forecast
0.0250
0.0225
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
2016-12-01
2016-12-05
2016-12-09
2016-12-13
2016-12-17
2016-12-21
2016-12-25
2016-12-29
2017-01-01