The purpose of this project was to build and evaluate a working forecasting application using real hotel demand data. I created a forecasting app in Hugging Face Spaces that allows users to upload a dataset, configure forecasting settings, run classical models, and export evaluation metrics, cross-validation results, and forward forecasts. For my analysis, I used the daily occupancy data for hotel_0 from the sample_hotels dataset and focused on generating a short-term 14-day forecast using the Naive model. The Naive model was chosen because it provides a benchmark for performance: it predicts that the next value will be equal to the most recent observed value. This makes it simple, fast, transparent, and reliable for understanding the baseline behavior of the time series.

To begin, I loaded the hotels_input.csv file into the application. The interface displayed the model configuration options, including frequency, evaluation strategy, backtest horizon, step size, number of windows, and future horizon. I selected a fixed-window evaluation strategy with a 14-day backtest horizon, a 7-day step size, 3 windows, and a 14-day forecast horizon. Only the Naive model was activated so that the results would be easy to interpret and so that the computation would complete quickly and smoothly. Insert Screenshot of App Interface Here.

Once the model was executed, the application produced a backtesting plot comparing historical values with the Naive model's predictions. Because the dataset contains many hotels and the Naive model produces the same forecast across the horizon, the backtest figure appears extremely dense when all unique_ids are plotted. The figure shows a mass of overlapping lines representing each hotel's actual values and their corresponding Naive predictions. While visually crowded, this confirms the app was able to compute forecasts for the entire dataset. Insert Backtest Plot Screenshot Here.

Below the plot, the evaluation metrics and cross-validation results were displayed. For the Naive model, the evaluation returned a bias score for each hotel. For example, hotel_0 produced a bias value of approximately 0.0326, indicating a very small average over-forecast of about three percent. Many hotels showed bias values close to zero, which means the Naive model was generally not systematically over- or under-predicting. The cross-validation table provided actual versus predicted values for the backtest period. For hotel_0, the cutoff date was June 2, 2023, and the Naive forecast for the following dates (June 3 through June 11) repeated the last observed value, 0.691358. Actual occupancy fluctuated around this number, giving a clear sense of how well the Naive model performed during the backtest window. Insert Screenshot of Evaluation Metrics and CV Results Here.

The application then generated a 14-day future forecast. As expected, all future predicted values for hotel_0 were identical because the Naive model uses the most recent observation as the forecast for every future date. The forecast CSV showed predictions such as 0.82716 for dates beginning on July 1, 2023. This behavior is consistent with the Naive model and provides an easy-to-interpret baseline forecast.

Overall, the app successfully loaded the dataset, ran the Naive model, computed evaluation metrics, produced cross-validation results, generated future forecasts, and exported all outputs.

The Naive model's performance was stable and reasonable, showing minimal bias and producing interpretable results. Although simple, the model provided a strong foundation for understanding the behavior of the dataset and verifying that the forecasting pipeline was working correctly. This completes the forecasting task and demonstrates successful deployment and analysis using the ISA444 forecasting application.