TASK2

Technical Report:

* I find that Here I only going to use Univariate feature to Forecast ,so I train my Model with Exponential smoothing algorithm.
* Exponential smoothing is easy to implement and computationally efficient. It's particularly useful for forecasting when there's a stable pattern in the data and random variation around that pattern.

CODE:

from statsmodels.tsa.statespace.exponential\_smoothing import ExponentialSmoothing

model = ExponentialSmoothing(data\_filtered['MyWay'], trend='additive', seasonal=12)

model\_fit = model.fit()

# Forecast for May 1-7, 2024

forecast\_start = '2024-05-01'

forecast\_end = '2024-05-07

forecast = model\_fit.forecast(len(pd.date\_range(start=forecast\_start, end=forecast\_end))

print('Exponential Smoothing Forecast:')

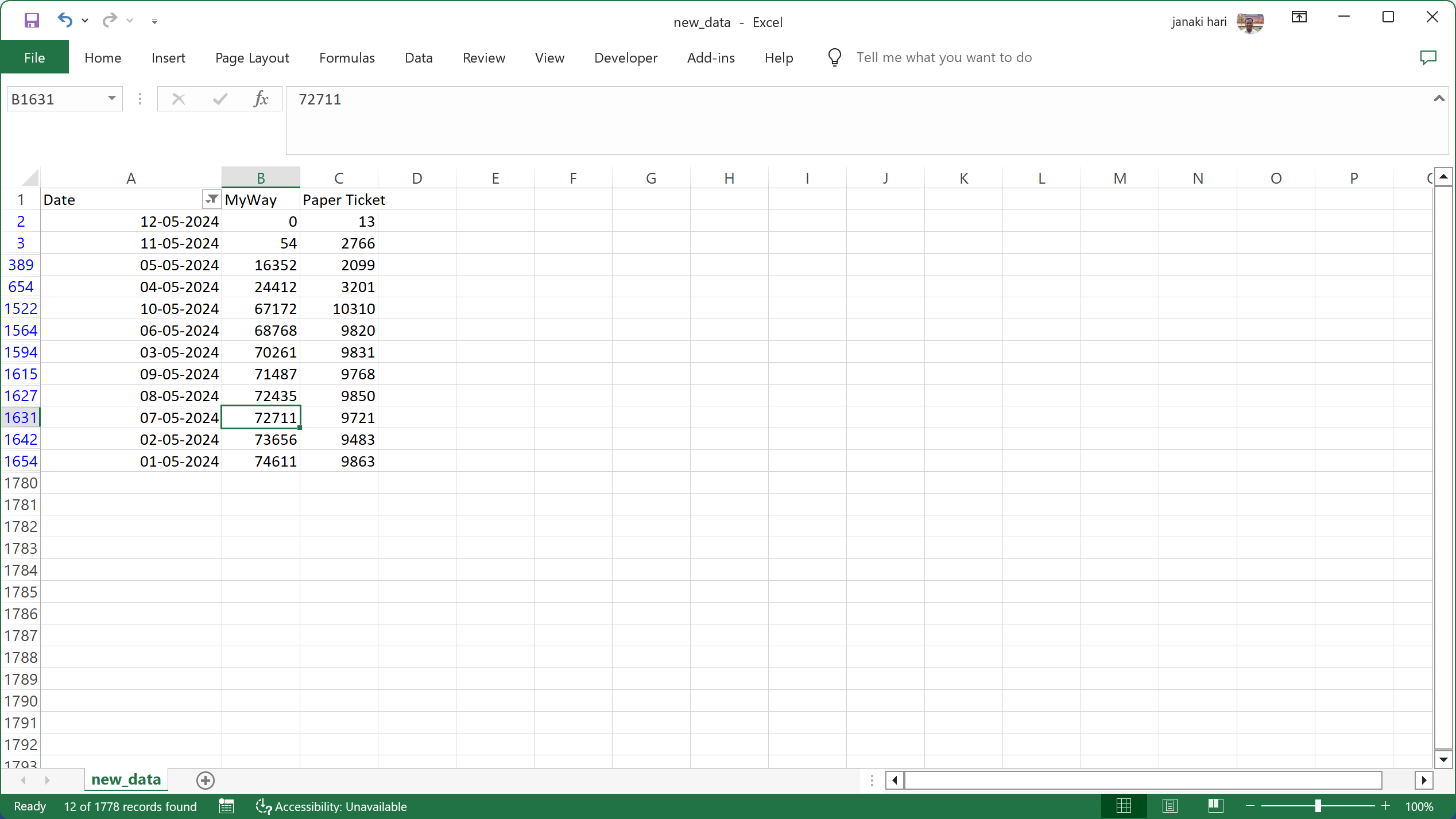
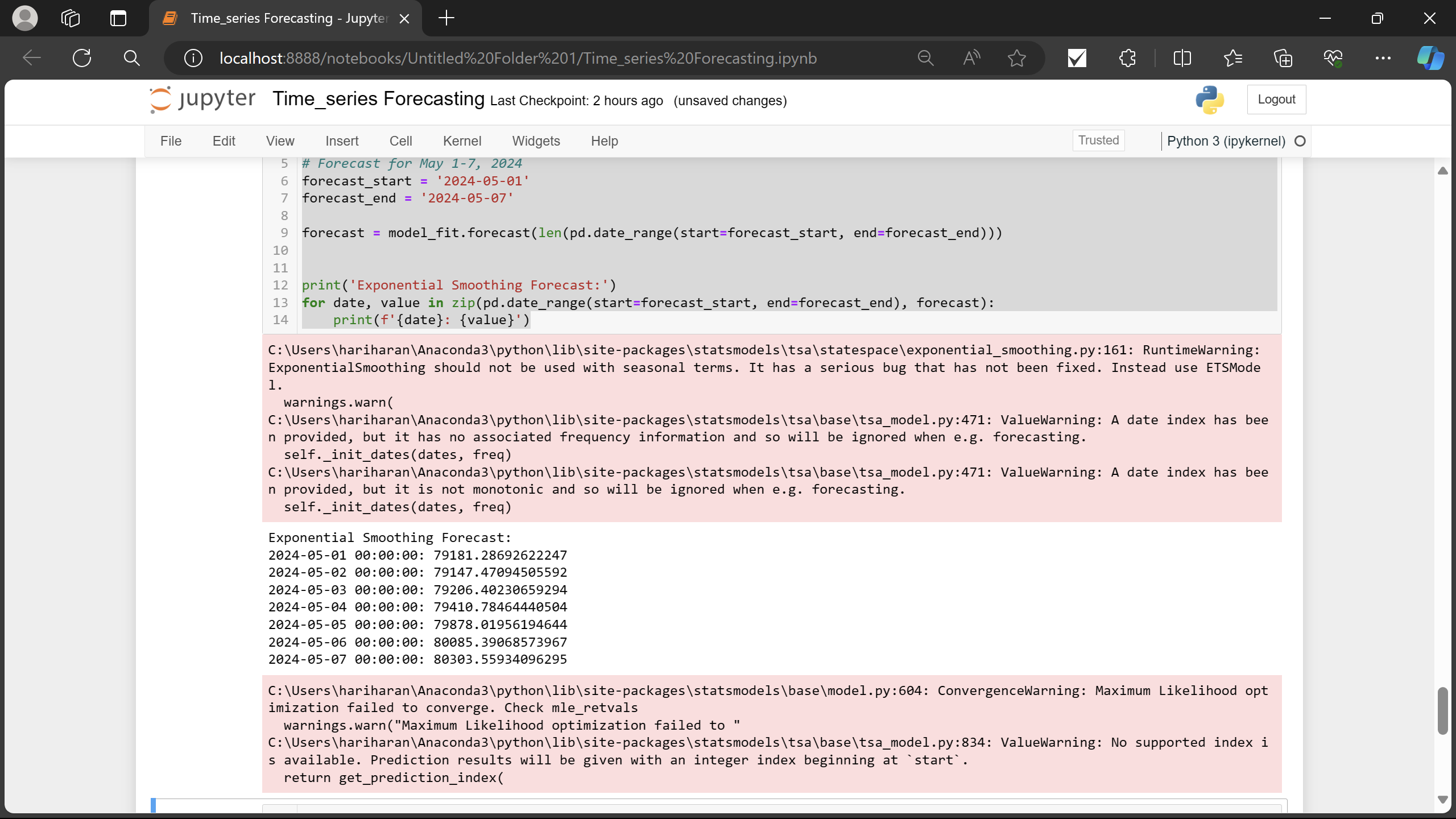
for date, value in zip(pd.date\_range(start=forecast\_start, end=forecast\_end), forecast):

print(f'{date}: {value}')

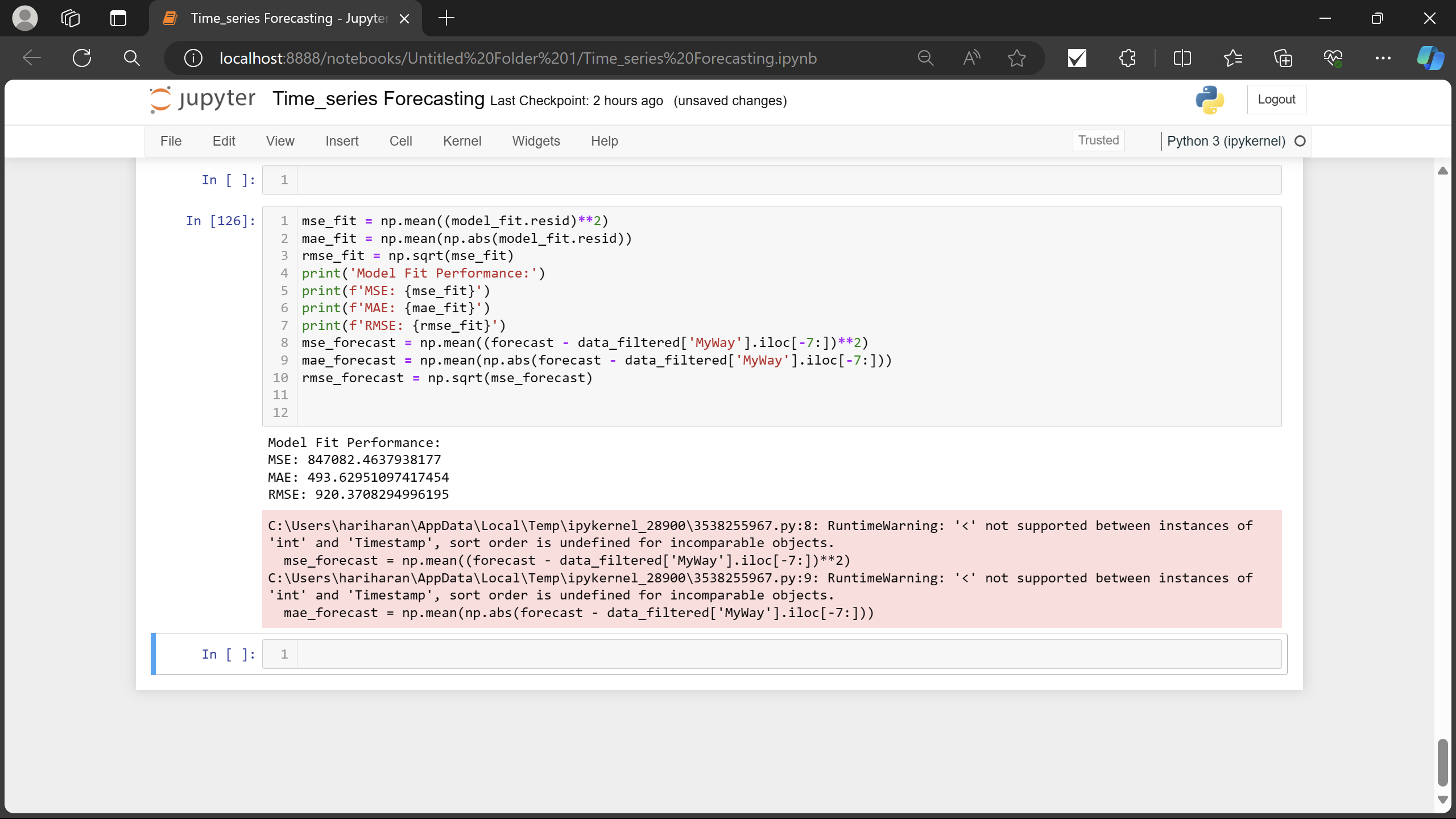
EXPLAINATION:

* First I import Exponential smoothing From statsmodels.tsa.statespace.exponential\_smoothing
* Stored the **Forecast \_start and Forecast\_end** in different Variable
* Train the Model with the [**MyWay] feature with Trend and Season**
* And Finally I **forecast with Forecast \_start and Forecast\_end** with help of trained Model

**OUTPUT:** **Actual Output:**

****

**Perfomance Metrics:**

****

**Model Fit Performance:**

These metrics suggest that the model fit is good. The MSE and RMSE are both relatively high, and the MAE is also relatively high. This means that the predicted values are not very close to the actual values on average.

There are a few possible reasons for this. One possibility is that the model is not complex enough to capture the underlying structure of the data. Another possibility is that the **data is too noisy and the model** is not able to learn the true relationships between the variables.