**Time Series Forecasting With Prophet**

**Resources :**

[**https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/**](https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/)

<https://facebook.github.io/prophet/>

<https://facebook.github.io/prophet/docs/quick_start.html>

<https://research.facebook.com/blog/2017/02/prophet-forecasting-at-scale/>

**Gemini**

**Time Series Forecasting with Prophet**

**Prophet** is a popular open-source forecasting tool developed by Facebook. It's designed to be easy to use, even for users with limited statistical knowledge. Prophet is particularly well-suited for time series data with strong seasonality and trend components.

**Key Features of Prophet**

* **Flexibility:** Easily handles various time series patterns, including seasonality, trend, holidays, and outliers.
* **Robustness:** Handles missing data and changes in trend or seasonality.
* **Interpretability:** Provides clear explanations of the model's predictions, making it easy to understand the factors driving forecasts.

**Components of a Prophet Model**

* **Trend:** The overall growth or decline over time.
* **Seasonality:** Repeating patterns within a specific time period (e.g., daily, weekly, yearly).
* **Holidays:** Effects of specific dates or events.
* **Regressors:** Additional covariates that can influence the forecast.

**Basic Usage**

1. **Import the Prophet library:**

Python

from prophet import Prophet

1. **Prepare your data:**
   * Create a pandas DataFrame with two columns: ds (date) and y (value).
   * Ensure the ds column is a pandas datetime object.
2. **Create a Prophet model:**

Python

model = Prophet()

1. **Fit the model to your data:**

Python

model.fit(df)

1. **Make predictions:**
   * Create a new DataFrame with a future date range:

Python

future = model.make\_future\_dataframe(periods=365)

* + Make predictions:

Python

forecast = model.predict(future)

**Example**

Python

import pandas as pd

from prophet import Prophet

# Create a sample DataFrame

df = pd.DataFrame({'ds': pd.date\_range(start='2020-01-01', end='2023-12-31'),

'y': [100 + 5\*i + 20\*np.sin(2\*np.pi\*i/365) for i in range(1461)]})

# Create a Prophet model

model = Prophet()

# Fit the model

model.fit(df)

# Make predictions for the next year

future = model.make\_future\_dataframe(periods=365)

forecast = model.predict(future)

# Plot the forecast

model.plot(forecast)

**Additional Features**

* **Custom Seasonality:** Define custom seasonality periods (e.g., quarterly, monthly).
* **Changepoints:** Specify change points where the trend might change.
* **Outlier Detection:** Automatically detect and handle outliers.
* **Holiday Effects:** Include effects of specific holidays or events.
* **Uncertainty Intervals:** Obtain prediction intervals to quantify uncertainty.

Prophet is a powerful tool for time series forecasting, offering a balance of ease of use and flexibility. By understanding its components and features, you can effectively apply it to a wide range of time series problems.

ChatGPT:

\*\*Time Series Forecasting with Prophet\*\* is a powerful tool for forecasting time series data. Developed by Facebook (now Meta), Prophet is designed to handle various challenges common in time series forecasting, such as missing data, outliers, and seasonality. It is particularly user-friendly and effective for datasets with daily observations.

### Key Features of Prophet:

1. \*\*Additive model\*\*: Prophet decomposes time series data into three main components:

- \*\*Trend\*\*: The overall increase or decrease in the data over time.

- \*\*Seasonality\*\*: Periodic patterns that repeat over a known period (e.g., weekly, yearly).

- \*\*Holidays/Events\*\*: Special events that may affect the time series, such as holidays, promotions, etc.

2. \*\*Robust to outliers and missing data\*\*: Prophet can handle missing data well and is not too sensitive to outliers.

3. \*\*Automatic trend and seasonality detection\*\*: Prophet automatically detects trend changes and seasonality patterns, though you can manually set these if you have domain knowledge.

4. \*\*Works with different time frequencies\*\*: While designed for daily data, Prophet can also work with data at other time intervals (e.g., weekly, monthly).

5. \*\*Customizable\*\*: You can modify the model components (trend, seasonality, holidays) to suit your specific use case.

### How Prophet Works

The \*\*Prophet model\*\* fits a curve to the time series data by combining the trend, seasonality, and holiday components. It is based on the following additive model:

\[

y(t) = g(t) + s(t) + h(t) + \epsilon\_t

\]

Where:

- \( g(t) \) is the \*\*trend\*\* function modeling non-periodic changes.

- \( s(t) \) represents the \*\*seasonal\*\* component, capturing periodic effects (e.g., yearly seasonality).

- \( h(t) \) models the effects of \*\*holidays\*\* (optional).

- \( \epsilon\_t \) is the \*\*error term\*\* representing noise or irregular fluctuations.

### Steps to Forecast Time Series with Prophet

#### 1. \*\*Install Prophet\*\*

Prophet can be installed using `pip`:

```bash

pip install prophet

```

#### 2. \*\*Import Necessary Libraries\*\*

You’ll need `pandas` to manipulate your time series data and `Prophet` to create the forecasting model:

```python

import pandas as pd

from prophet import Prophet

```

#### 3. \*\*Prepare the Data\*\*

Prophet expects a `pandas` DataFrame with two columns:

- \*\*ds\*\*: The date column.

- \*\*y\*\*: The value you're trying to forecast.

Example:

```python

# Create example time series data

data = {'ds': ['2021-01-01', '2021-01-02', '2021-01-03', '2021-01-04'],

'y': [10, 12, 15, 13]}

df = pd.DataFrame(data)

df['ds'] = pd.to\_datetime(df['ds']) # Ensure the 'ds' column is datetime

```

#### 4. \*\*Fit the Prophet Model\*\*

Initialize and fit the model:

```python

# Initialize the Prophet model

model = Prophet()

# Fit the model on the dataset

model.fit(df)

```

#### 5. \*\*Make a Forecast\*\*

You need to create a future dataframe that tells Prophet the periods you want to predict. You can specify the number of periods for forecasting:

```python

# Create a dataframe for future dates

future = model.make\_future\_dataframe(periods=30) # Forecast 30 days into the future

# Make predictions

forecast = model.predict(future)

```

The `forecast` DataFrame will include predicted values (`yhat`), as well as the lower and upper uncertainty intervals (`yhat\_lower`, `yhat\_upper`).

#### 6. \*\*Plot the Forecast\*\*

Prophet has built-in plotting functions to visualize the forecast:

```python

# Plot forecast

model.plot(forecast)

```

You can also visualize the trend, seasonality, and holidays separately:

```python

# Plot components (trend, seasonality)

model.plot\_components(forecast)

```

### Customizing the Prophet Model

1. \*\*Adding Holidays\*\*: You can add special events or holidays to the model to account for anomalies in your time series.

```python

from prophet.plot import plot\_forecast\_component

holidays = pd.DataFrame({

'holiday': 'special\_event',

'ds': pd.to\_datetime(['2023-12-25', '2024-01-01']),

'lower\_window': 0,

'upper\_window': 1

})

model = Prophet(holidays=holidays)

model.fit(df)

```

2. \*\*Handling Seasonality\*\*: You can specify whether Prophet should consider weekly, yearly, or custom seasonality.

```python

model = Prophet(yearly\_seasonality=True, weekly\_seasonality=True)

model.fit(df)

```

3. \*\*Trend Changepoints\*\*: Prophet automatically detects changepoints in your data where the trend changes. You can also provide custom changepoints if you know when these shifts occurred.

### Example: Simple Daily Sales Forecast

```python

import pandas as pd

from prophet import Prophet

# Load your time series data

df = pd.read\_csv('daily\_sales.csv') # Assume your file has 'ds' and 'y' columns

df['ds'] = pd.to\_datetime(df['ds']) # Convert 'ds' to datetime format

# Initialize the model

model = Prophet()

# Fit the model to the data

model.fit(df)

# Create future dates (let's say 30 days into the future)

future = model.make\_future\_dataframe(periods=30)

# Make the forecast

forecast = model.predict(future)

# Plot the forecast

model.plot(forecast)

```

### Common Use Cases:

1. \*\*Sales forecasting\*\*

2. \*\*Demand forecasting\*\*

3. \*\*Traffic prediction for websites\*\*

4. \*\*Temperature forecasting\*\*

### Advantages of Prophet:

- \*\*Ease of use\*\*: Simple interface, minimal setup required.

- \*\*Handles missing data and outliers\*\* effectively.

- \*\*Domain-specific knowledge\*\* can be incorporated by adjusting seasonality and holidays.

- \*\*Automated changepoint detection\*\* for capturing trend shifts.

### Disadvantages:

- It may \*\*not work well for highly complex or non-linear datasets\*\*, where more advanced techniques like LSTMs or ARIMA models might perform better.

- For very high-frequency data (e.g., hourly), Prophet may require careful tuning.

Would you like a more detailed code example or further guidance on using Prophet for your specific dataset?c