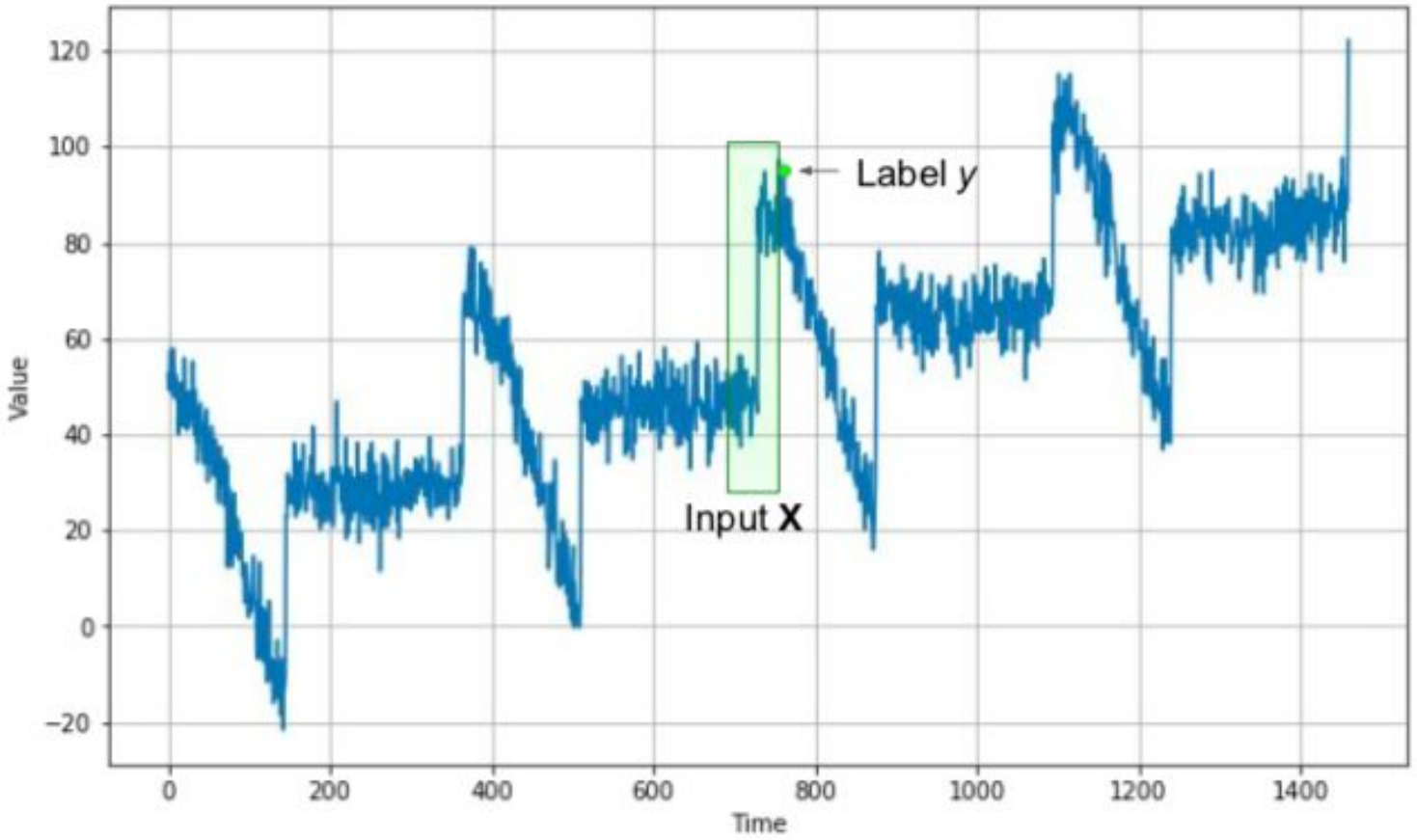
Time Series Models

Machine Learning on Time Windows



Grain of salt …

“Forecasts may tell you a great deal about the forecaster; they tell you nothing about the future.”

— Warren Buffett

Time Series (TS)

- Time Series are a type of Sequence Model
- Ordered sequence of values at usually equally spaced amount of time
- Multi value per time = multivariate

Time Series (TS)

- Time Series pertains to the sequence of observations collected in constant time intervals, be it daily, monthly, quarterly or yearly.
- Time Series Analysis involves developing models used to describe the observed time series and understand the "why" behind its dataset. This involves creating assumptions and interpretations about a given data.
- Time Series Forecasting makes use of the best-fitting model essential to predicting future observations based on the complex processing of current and previous data.

TS Use Cases

- Imputation – fill data gaps, even before start
- Detect anomalies
- Sound wave ts to detect words with a NN
- Stock market time series analysis
- Medical monitoring

TS Methods

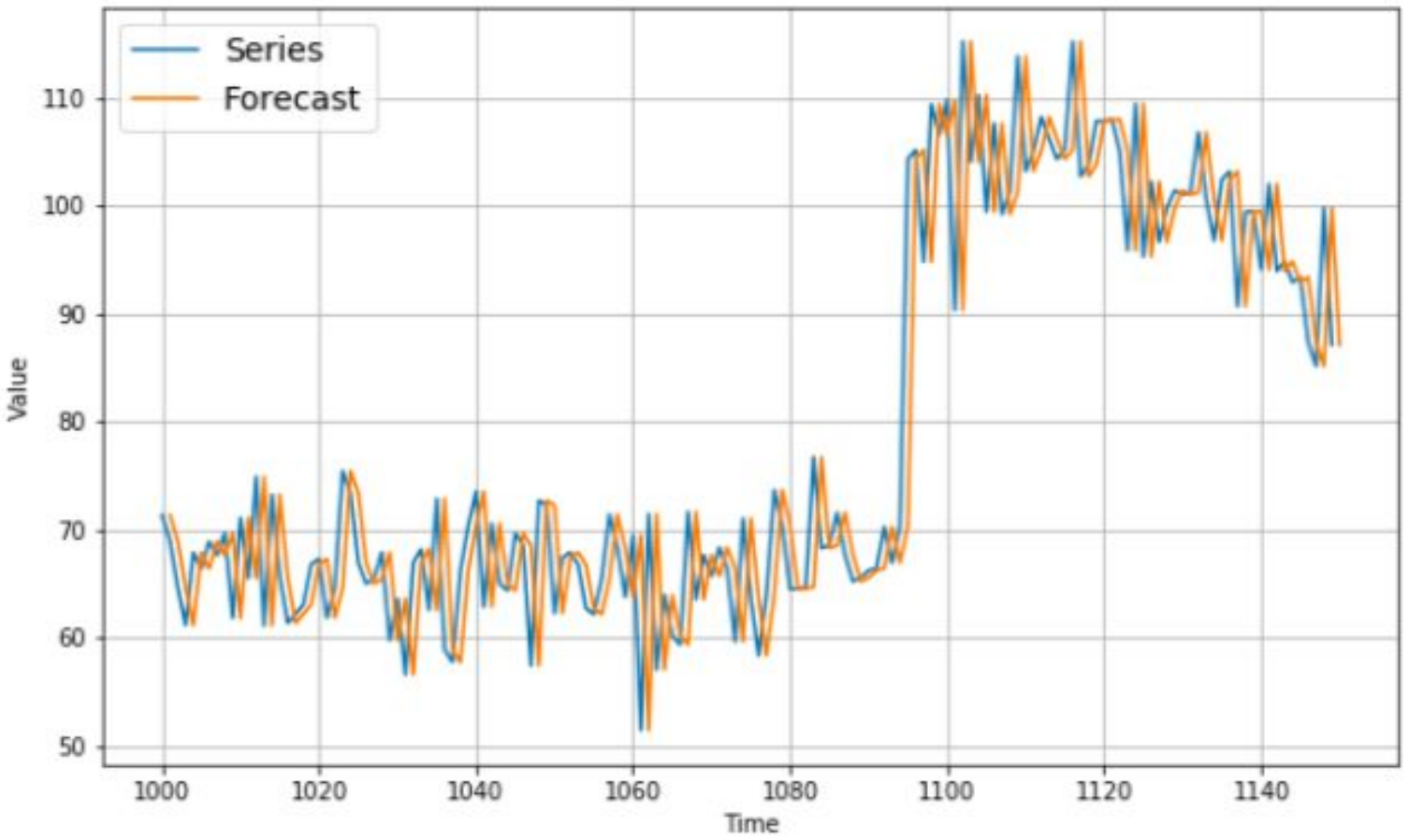
- **TS forecasting**: the most common practice in time series analysis. Given a time series, these techniques aim at predicting future values with a certain confidence interval using a computational model.
- **TS classification**: given a time series, these techniques aim at extracting relevant aggregated features from the time series in order to determine whether it belongs to a certain class. An example of time series classification is the analysis of ECG recordings to understand whether they are taken from a healthy or sick patient. These methods are more akin to standard ML classification.

CLASSICAL (STATISTICAL) TS ALGOS

Classical TS Models

- Naïve model
  - In most cases, naïve models are applied as a random walk (with the last observed value used as a unit for the next period forecast) and a seasonal random walk (with a value from the same period of the last observed time span used as a unit of the forecast).

Naive Forecasting



Exponential Smoothing Model

- The method uses the foundation of machine learning time series classification. Forecasts are made on the basis of captured weighted averages and according to weights decreasing as the observer tracks back in time.
- The method implies a smoothing constant that helps make previous data less essential and the newest data more crucial to build time-series predictions. The constant adjusts when building forecasts to make more recent data more or less valuable compared to historical data. The model can not help detect trends and seasonality. However, a number of extensions of the simple exponential smoothing (SES) have been introduced to include the trend/damped trend and seasonality.

ARIMA/SARIMA

- ARIMA stands for the combination of Autoregressive (AR) and Moving Average (MA) approaches within building a composite model of the time series. ARIMA models include parameters to account for season and trend (for instance, dummy variables for weekdays and they're distinguishing). In addition, they allow for the inclusion of autoregressive and moving average terms to handle the autocorrelation embedded in the data.

- SARIMA stands for Seasonal Autoregressive Integrated Moving Average: it widens the application of the ARIMA by including a linear combination of past seasonal values and/or forecast errors.

Linear Regression method

- Linear regression is the simple statistical technique commonly used for predictive modeling. Breaking it down to basics, it comes to providing an equation of independent variables on which our target variable is built upon. The defined relationship between variables helps predict future values.

ML TS ALGOS

Multi-Layer Perceptron (MLP)

- As an applied machine learning approach, the time series model forecasting implies the triple structure of the initial layer of the network, which takes in an input, a hidden layer of nodes, and an output layer used to make a prediction.

RNN, LSTM

- Recurrent Neural Network (RNN)
  - RNNs are basically neural networks with memory that can be used for predicting time-dependent targets. Recurrent neural networks can memorize the previously captured state of the input to make a decision for the future time step. Recently, lots of variations have been introduced to adapt Recurrent Networks to a variety of domains.
- Long Short-Term Memory (LSTM)
  - LSTM cells (special RNN cells) were developed to find the solution to the issue with gradients by presenting several gates to help the model make a decision on what information to mark as significant and what information to ignore. GRU is another type of gated recurrent network.

CNNs

- Besides methods mentioned above, Convolutional Neural Network models, or CNNs for short, as well as decision tree-based models like Random Forest and Gradient Boosting variations (LightGBM, CatBoost, etc.) can be applied to time series forecasting.

Evaluating Model Accuracy

- Speaking of time series methods, we should mention that visually identifying which machine learning model has the best accuracy is not always producing reliable results.
- Among the time series prediction models, the method of calculating the MAPE (Mean Absolute Percent Error) shows the best results as a quicker way to compare the overall forecast accuracy of a proposed model.
- The measures represent the percentage of average absolute error that occurred. The overall idea of calculations for the evaluation of model accuracy comes to the following proportion: the lower the MAPE, the better the forecast accuracy.

Metrics

```
errors = forecasts - actual

mse = np.square(errors).mean()

rmse = np.sqrt(mse)

mae = np.abs(errors).mean()

mape = np.abs(errors / x_valid).mean()
```

CLASSICAL VS ML TS MODELING

Classic vs ML TS Modeling

- Spyros Makridakis, et al. published a study in 2018 titled Statistical and Machine Learning forecasting methods: Concerns and ways forward."
- The study was a response to the increasing number of papers and claims that machine learning and deep learning methods offer superior results for time series forecasting with little objective evidence.
- Literally hundreds of papers propose new ML algorithms, suggesting methodological advances and accuracy improvements. Yet, limited objective evidence is available regarding their relative performance as a standard forecasting tool.
- As a response, the study includes eight classical methods and 10 machine learning methods evaluated using one-step and multiple-step forecasts across a collection of 1,045 monthly time series.
- Although not definitive, the results are intended to be objective and robust.

Data Sets Used

Time interval between successive observations	Types of time series data						
	Micro	Industry	Macro	Finance	Demographic	Other	Total
Yearly	146	102	83	58	245	11	645
Quarterly	204	83	336	76	57		756
Monthly	474	334	312	145	111	52	1428
Other	4			29		141	174
Total	828	519	731	308	413	204	3003

Algos Used

Classic Algos Used

- Naive 2, which is actually a random walk model adjusted for season.
- Simple Exponential Smoothing.
- Holt.
- Damped exponential smoothing.
- Average of SES, Holt, and Damped.
- Theta method.
- ARIMA, automatic.
- ETS, automatic.

ML Algos Used

- Multi-Layer Perceptron (MLP)
- Bayesian Neural Network (BNN)
- Radial Basis Functions (RBF)
- Generalized Regression Neural Networks (GRNN), also called kernel regression
- K-Nearest Neighbor regression (KNN)
- CART regression trees (CART)
- Support Vector Regression (SVR)
- Gaussian Processes (GP)
- Recurrent Neural Network (RNN)
- Long Short-Term Memory (LSTM)

Outcomes

- The study provides important supporting evidence that classical methods may dominate univariate time series forecasting, at least on the types of forecasting problems evaluated.
- The study demonstrates the worse performance and the increase in computational cost of machine learning and deep learning methods for univariate time series forecasting for both one-step and multi-step forecasts.
- These findings strongly encourage the use of classical methods, such as ETS, ARIMA, and others as a first step before more elaborate methods are explored, and requires that the results from these simpler methods be used as a baseline in performance that more elaborate methods must clear in order to justify their usage.
- It also highlights the need to not just consider the careful use of data preparation methods, but to actively test multiple different combinations of data preparation schemes for a given problem in order to discover what works best, even in the case of classical methods.
- Machine learning and deep learning methods may still achieve better performance on specific univariate time series problems and should be evaluated.
- More info here.

TS COMPONENTS

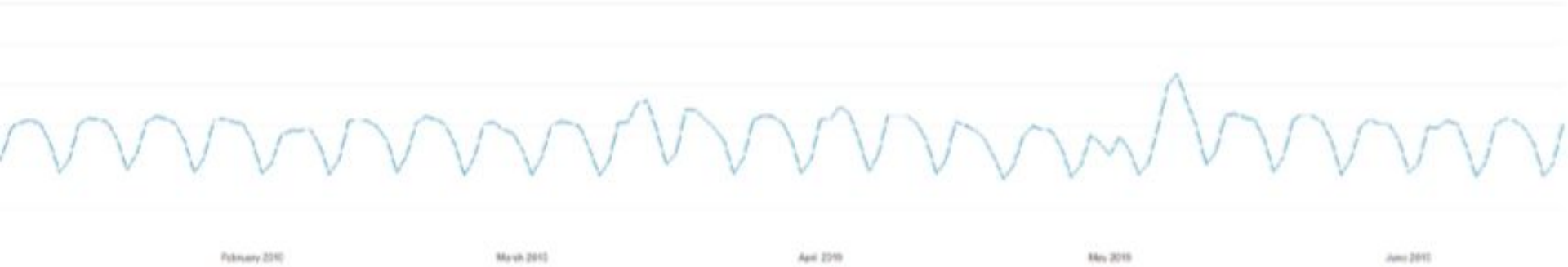
Components of Time Series data

- **Trends** - describe increasing or decreasing behavior of the time series frequently presented in linear modes
- **Seasonality** - highlight the repeating pattern of cycles of behavior over time
- **Irregularity/Noise** - regard the non-systematic aspect of time series deviating from the common model values
- **Cyclicity** - identify the repetitive changes in the time series and define their placement in the cycle.

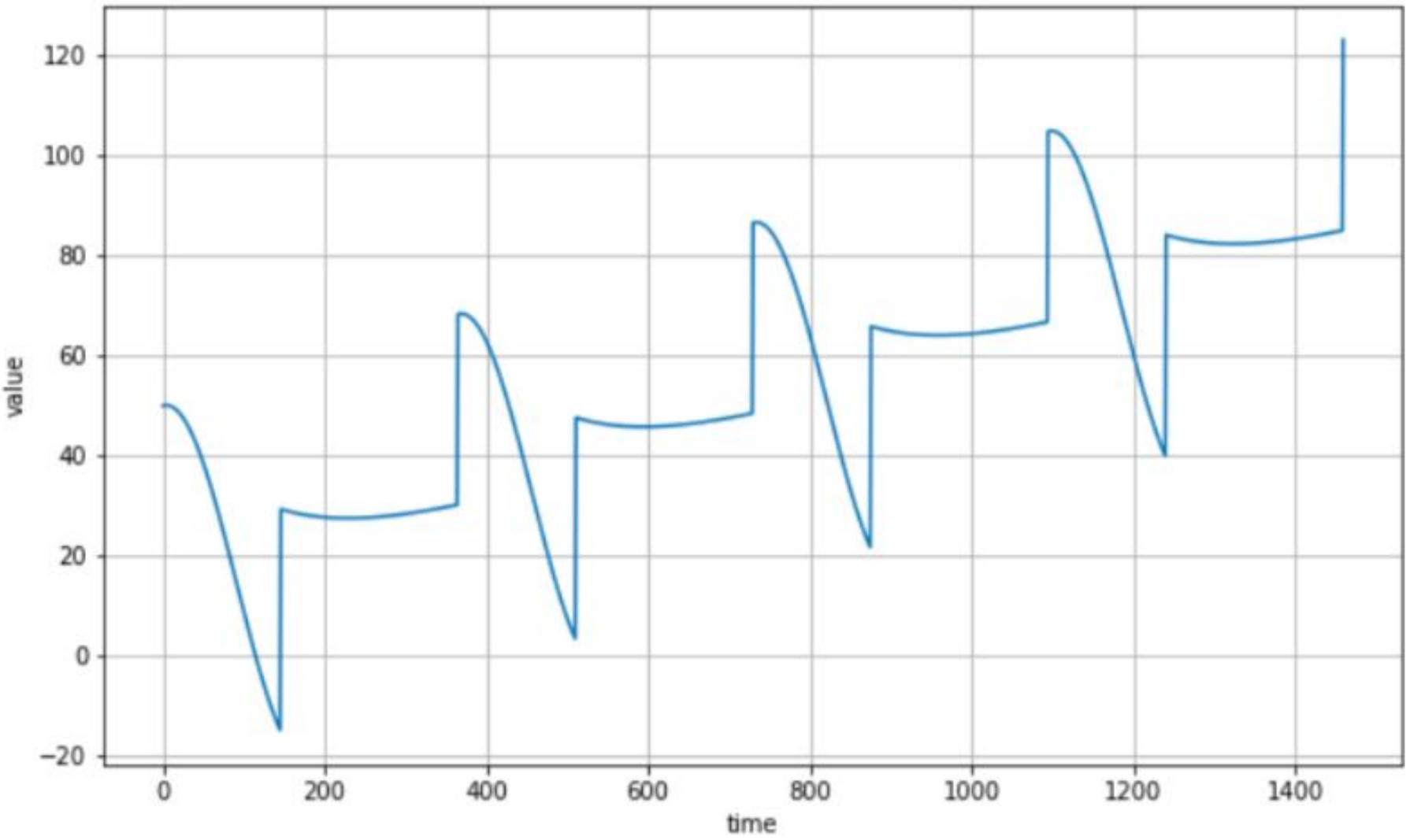
Trend


Density vs. Year
Density
Moore
100000000
10000000
1000000
100000
10000
1000
Density
Year
Chart Created by lmoroney@

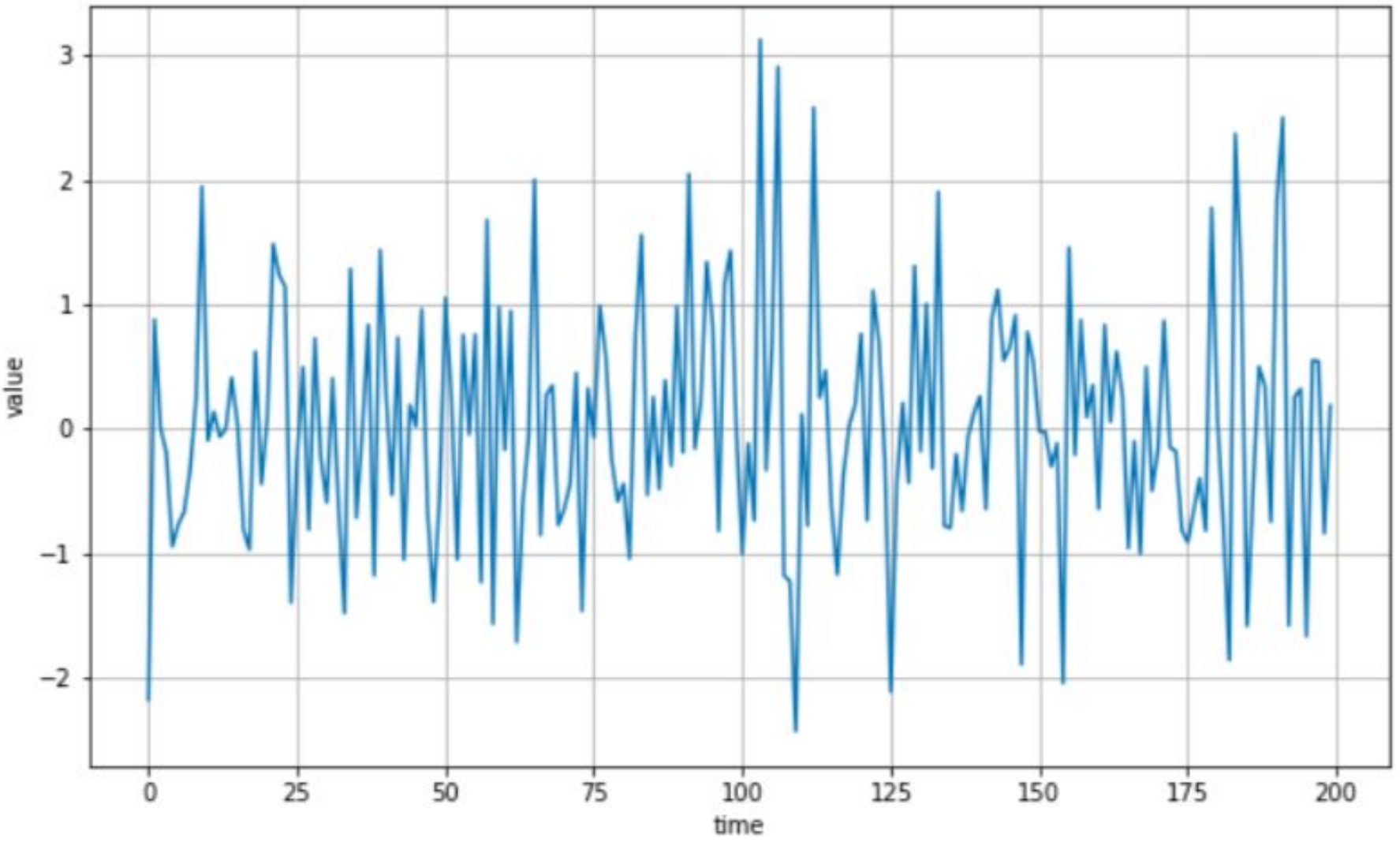
Seasonality



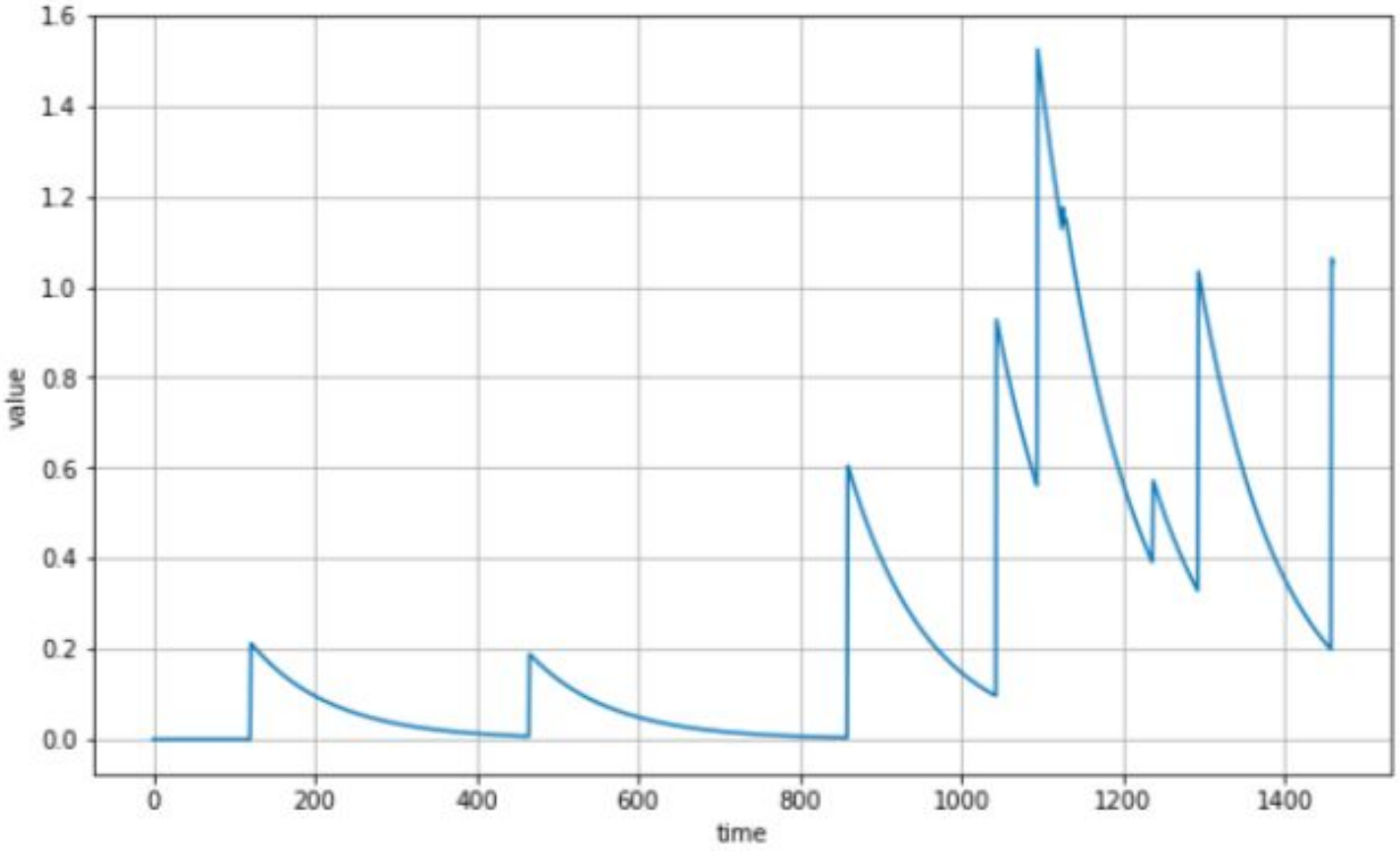
Trend and Seasonality



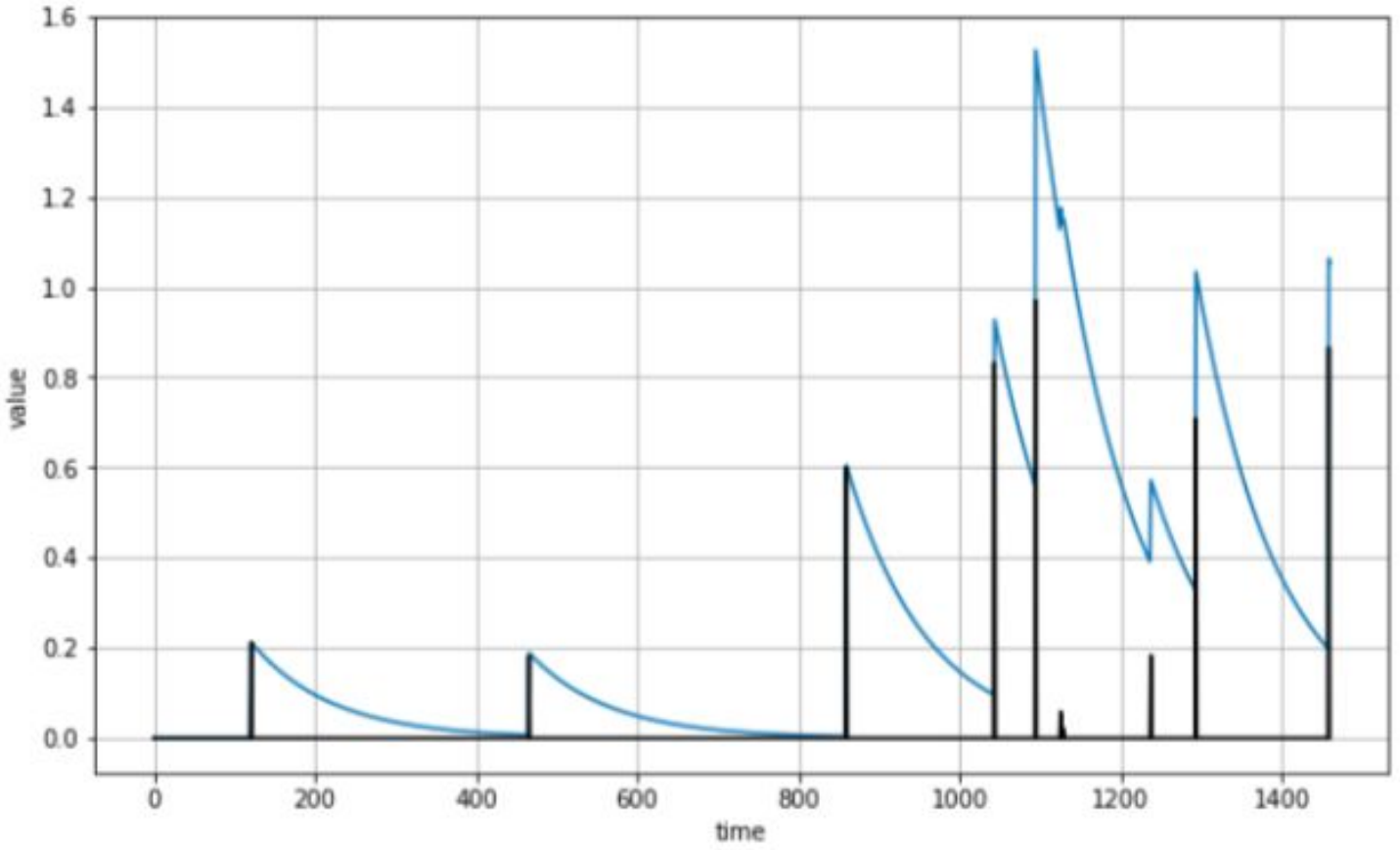
Noise



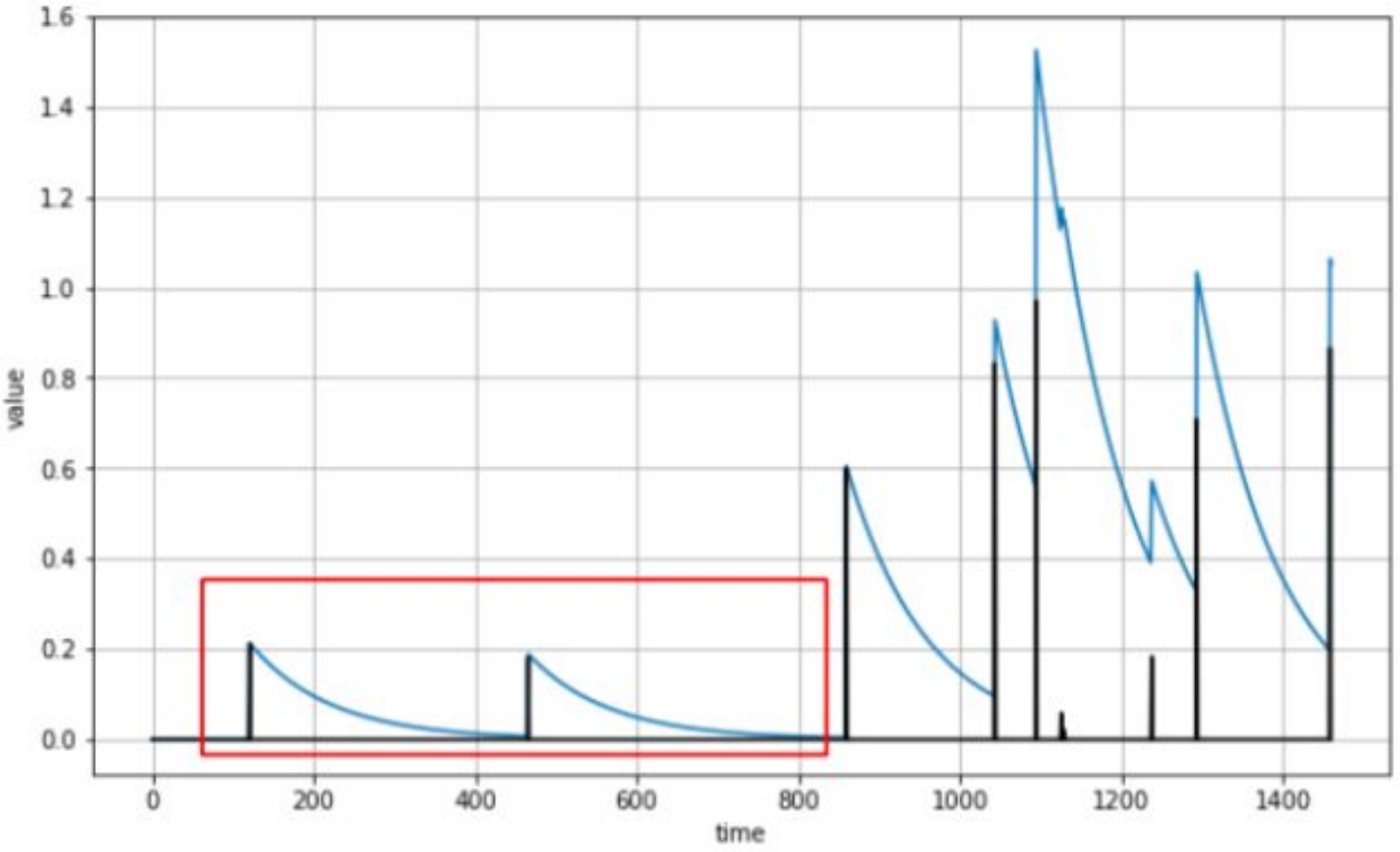
Autocorrelation



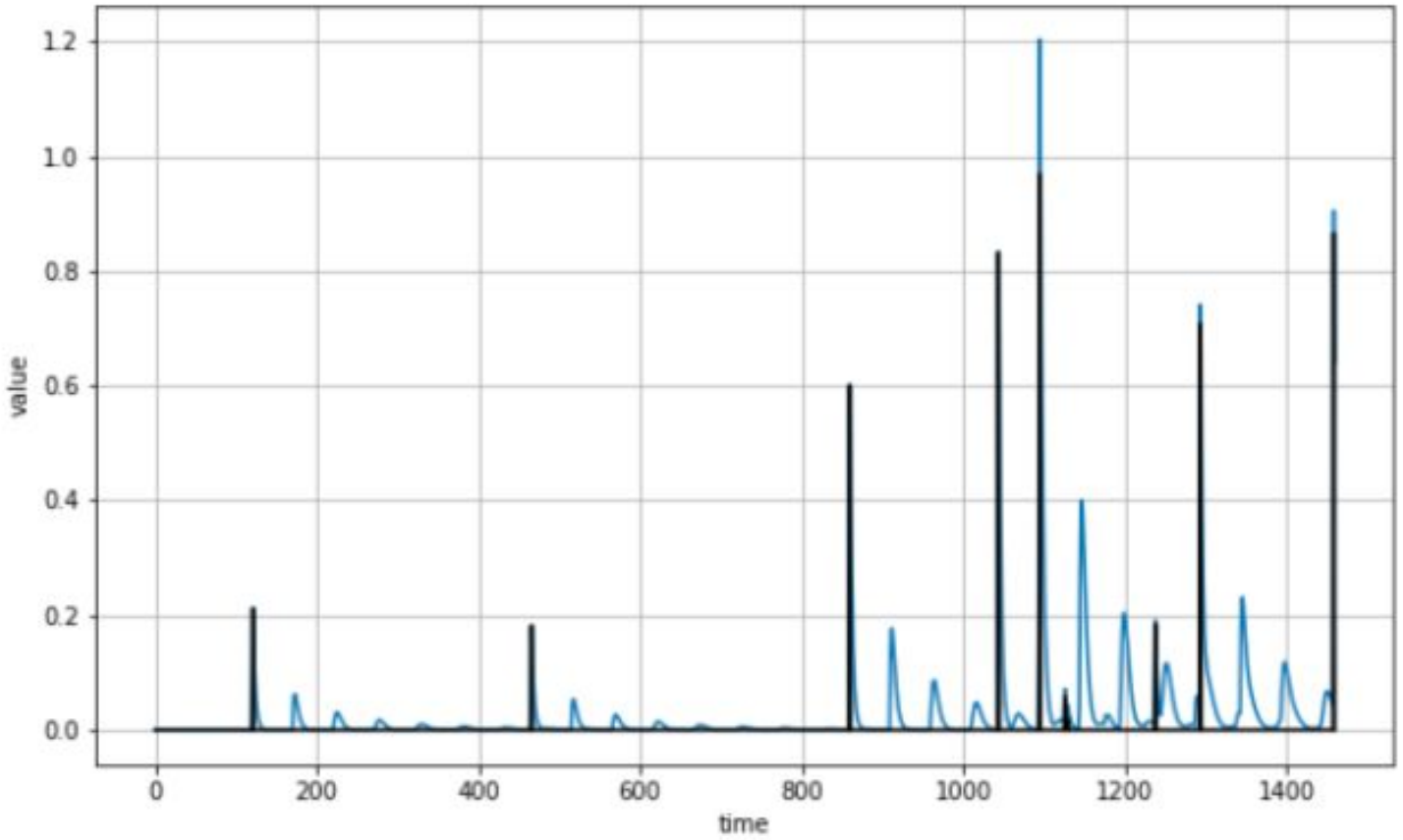
$v(t) = 0.99 \times v(t-1)$ + occasional spike



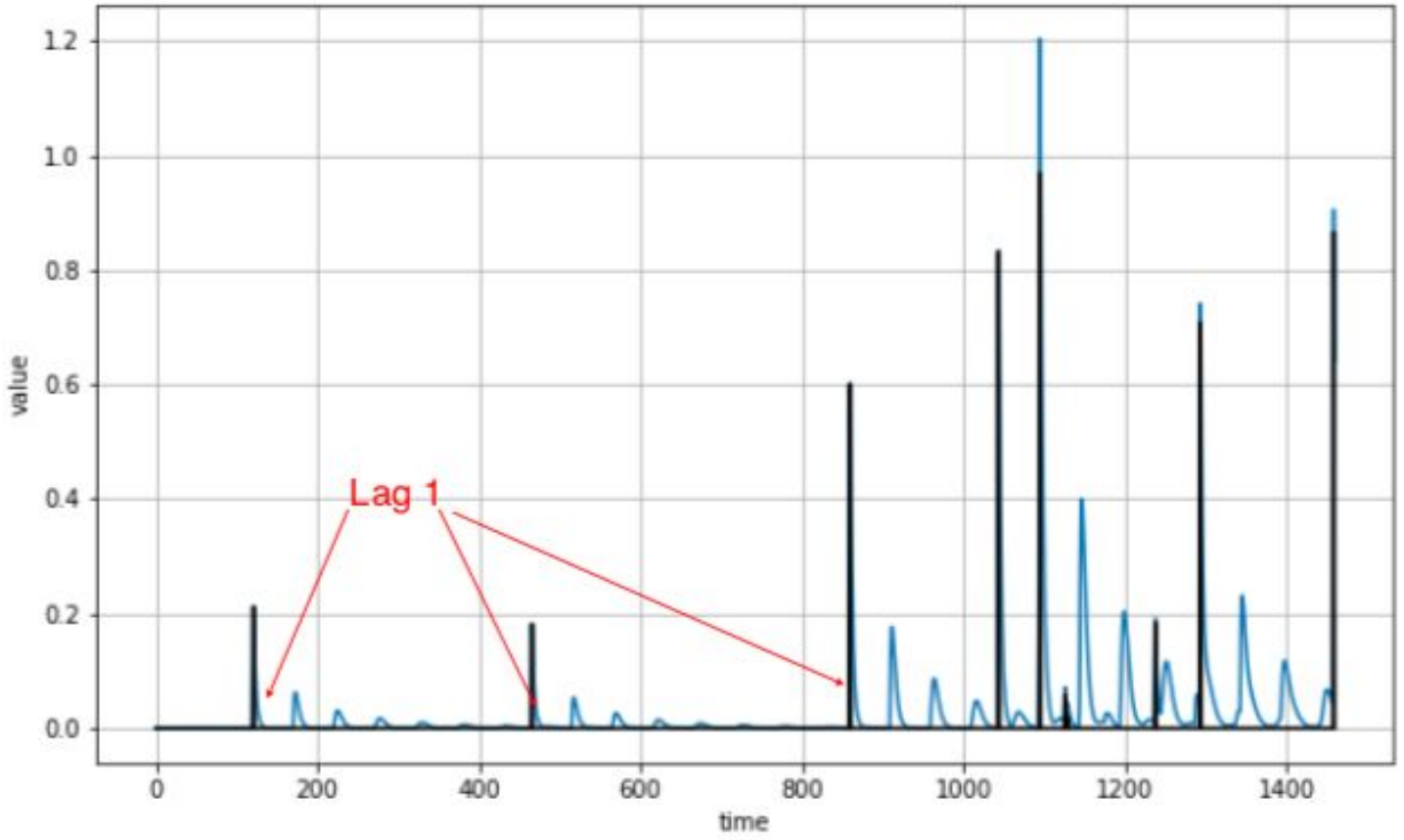
$v(t) = 0.99 \times v(t-1)$ + occasional spike



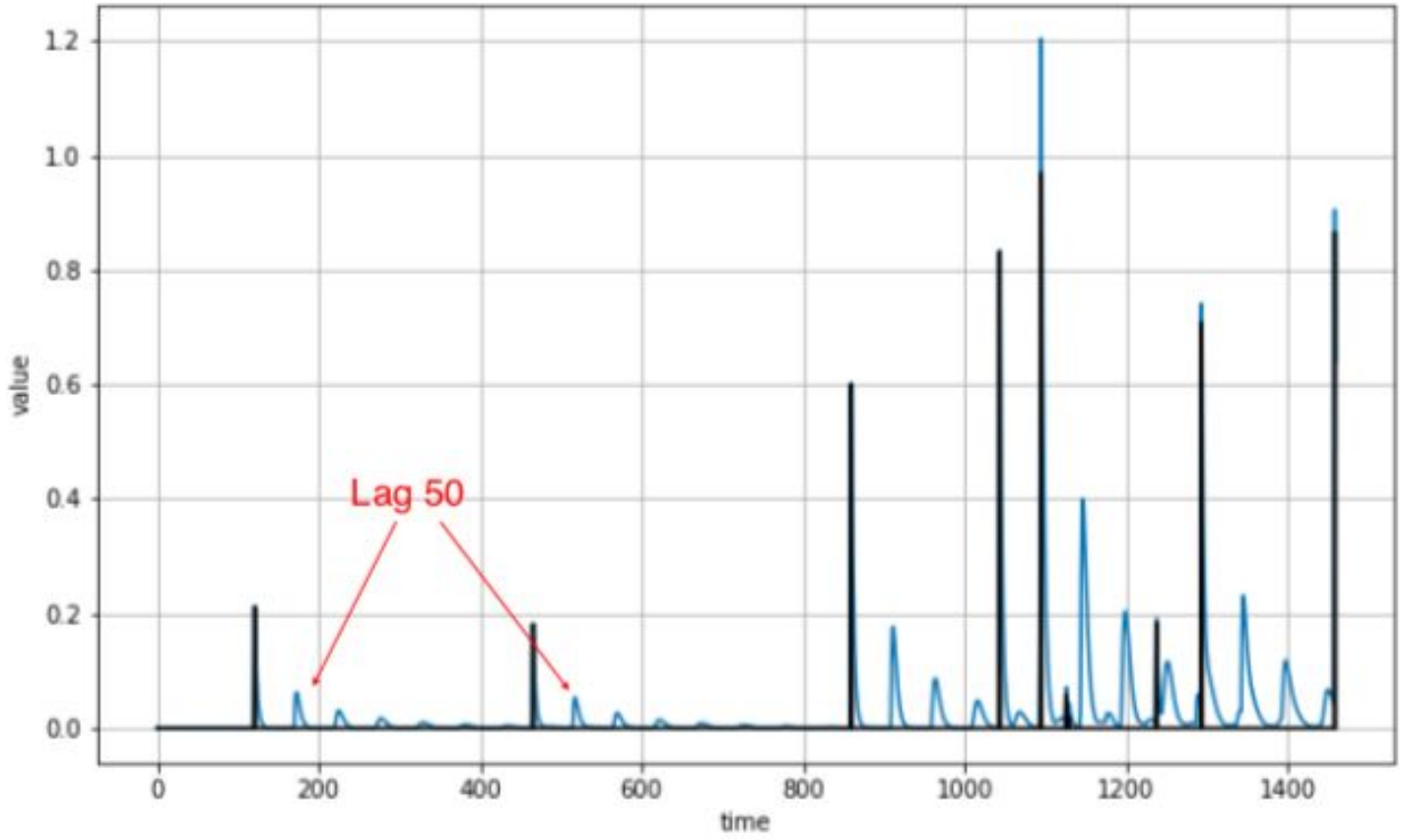
$$v(t) = 0.7 \times v(t-1) + 0.2 \times v(t-50) + \text{occasional spike}$$



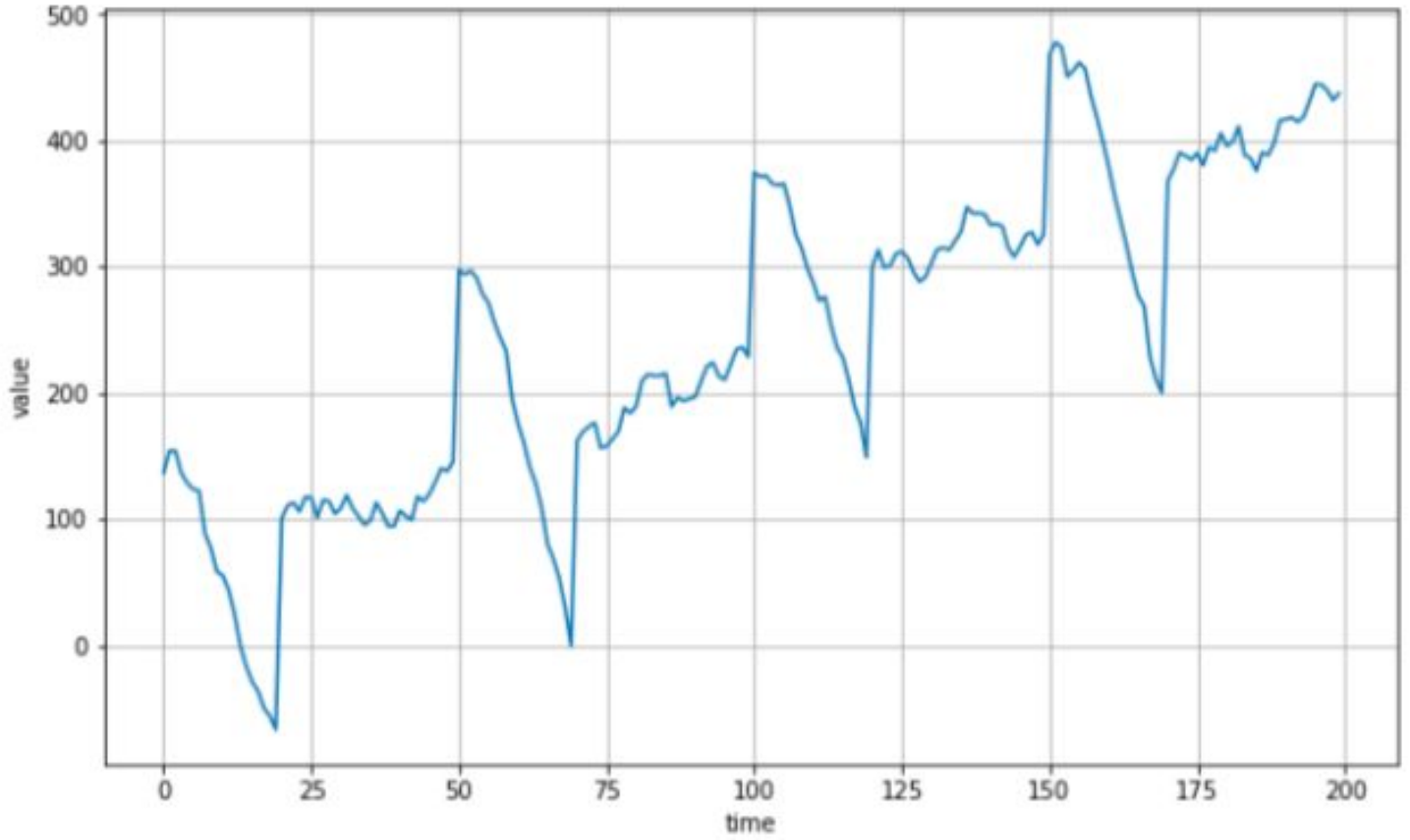
$$v(t) = 0.7 \times v(t-1) + 0.2 \times v(t-50) + \text{occasional spike}$$

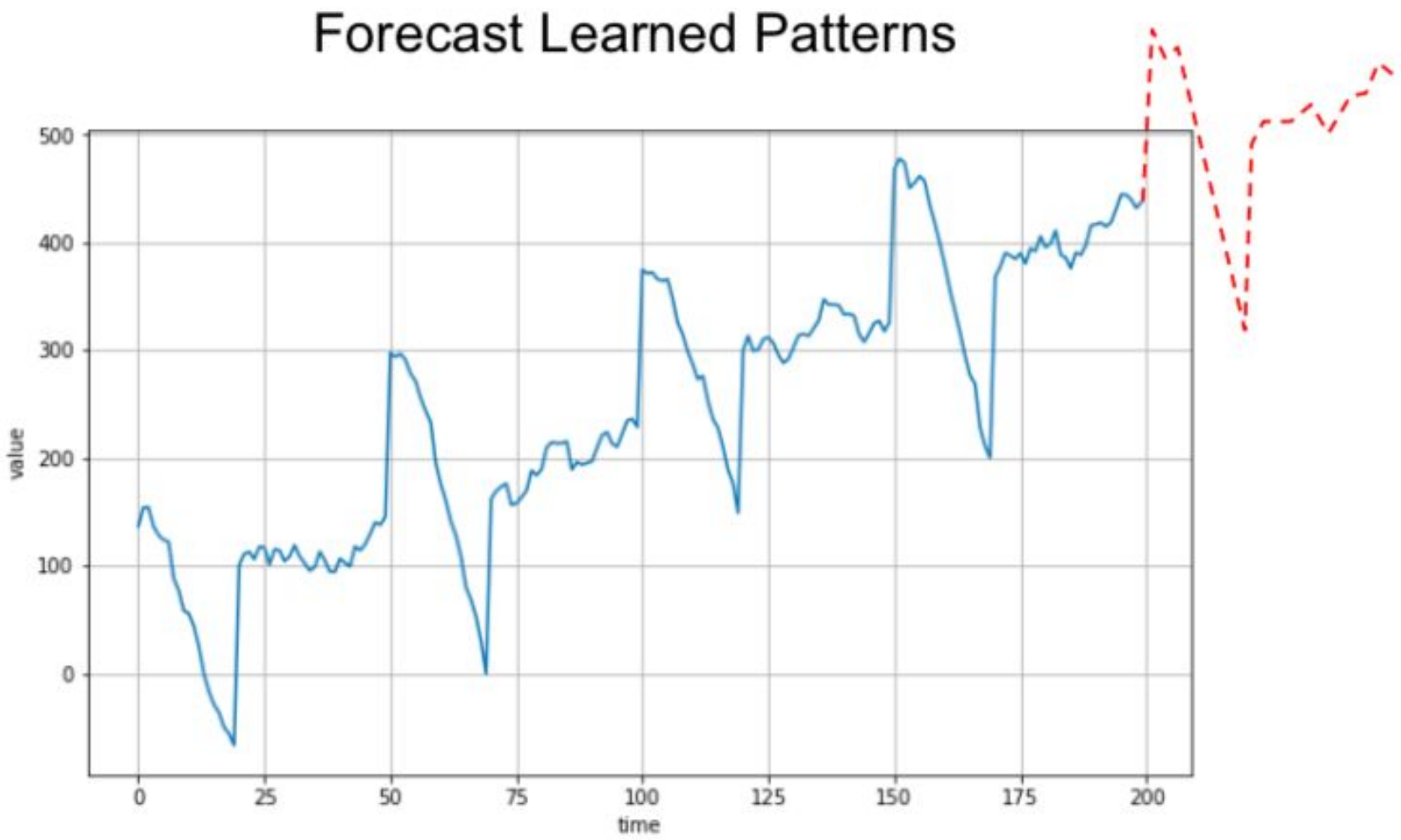


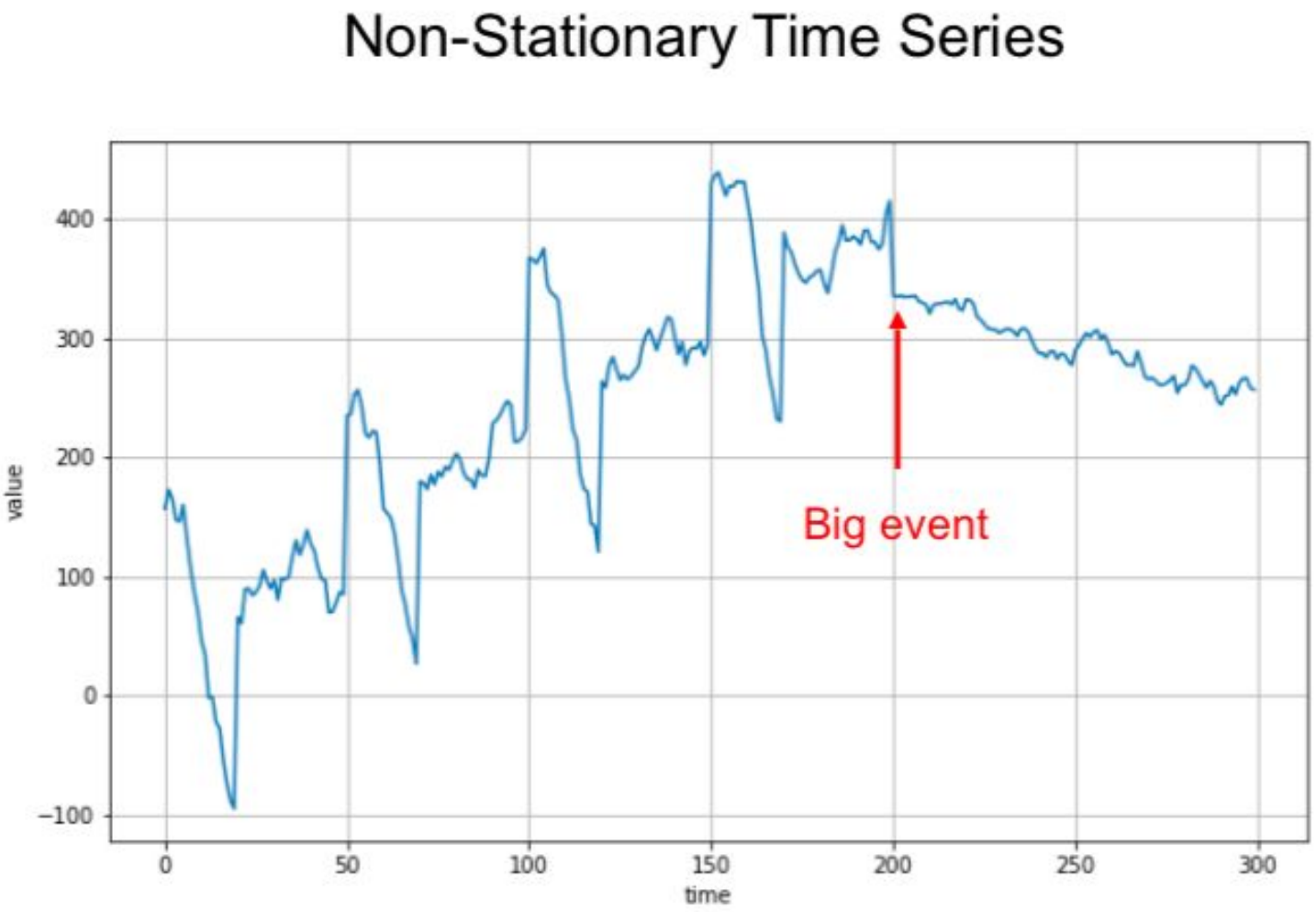
$$v(t) = 0.7 \times v(t-1) + 0.2 \times v(t-50) + \text{occasional spike}$$



Trend + Seasonality + Autocorrelation + Noise

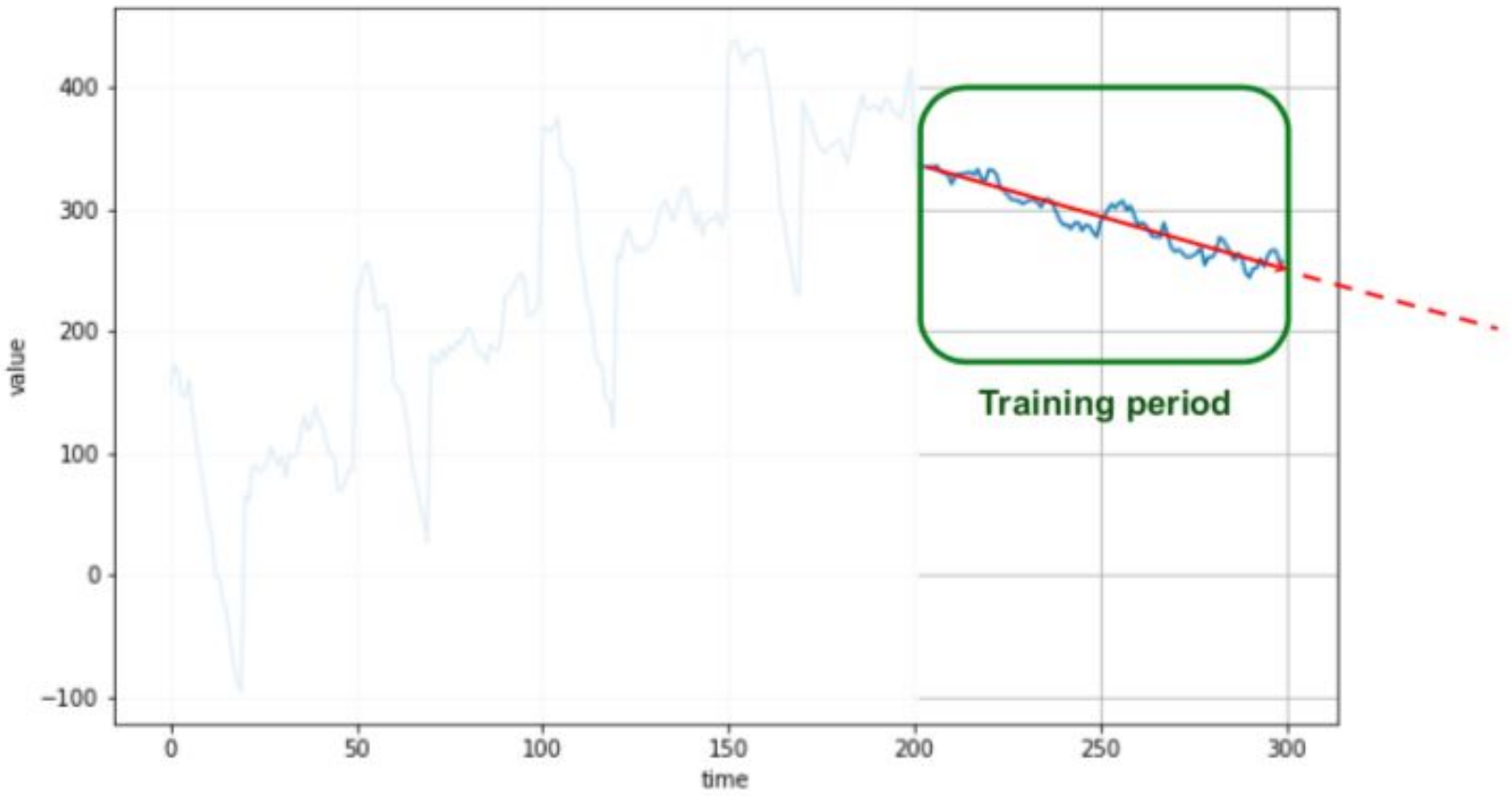


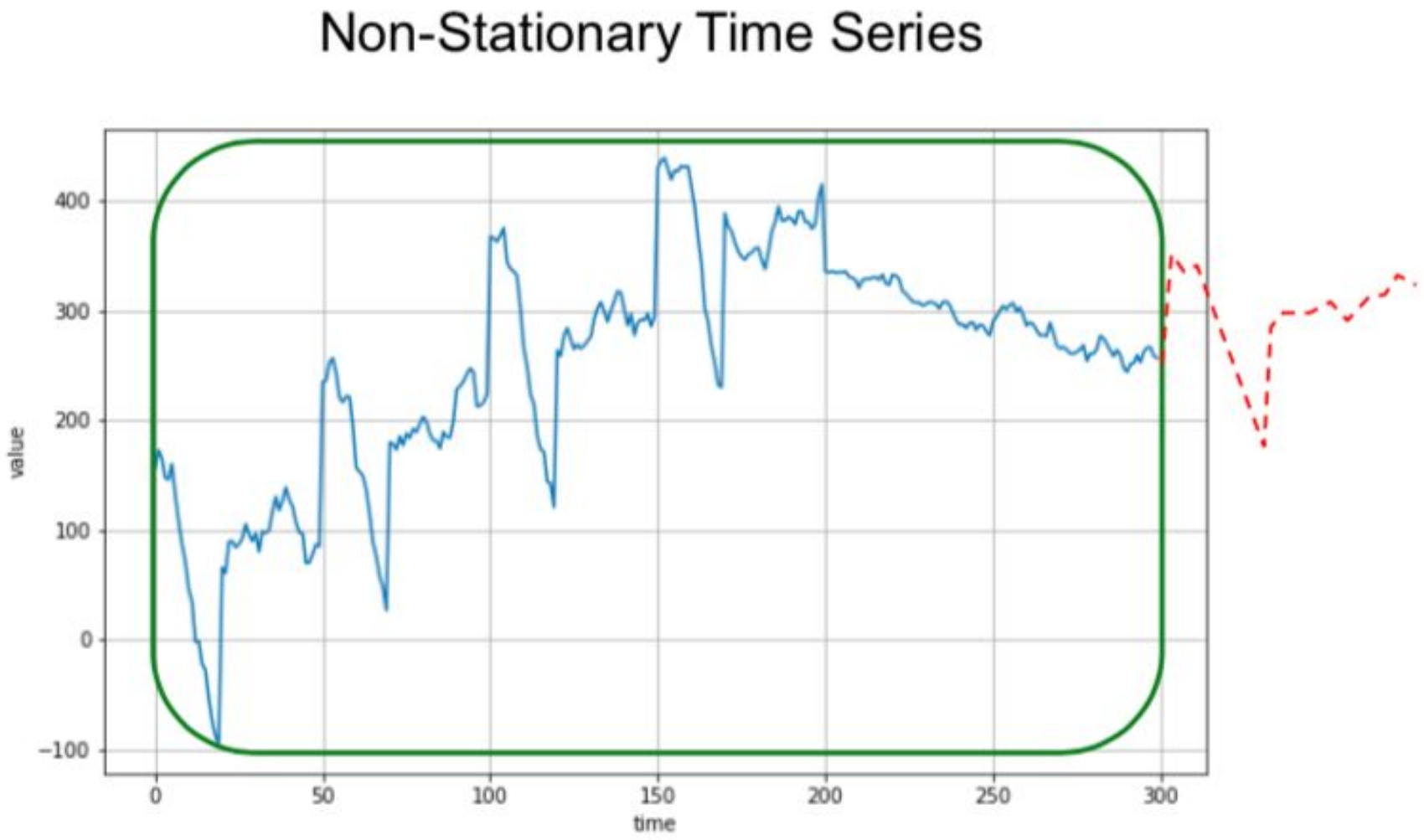

Forecast Learned Patterns
500
400
300
200
100
0
value
0
25
50
75
100
125
150
175
200
time



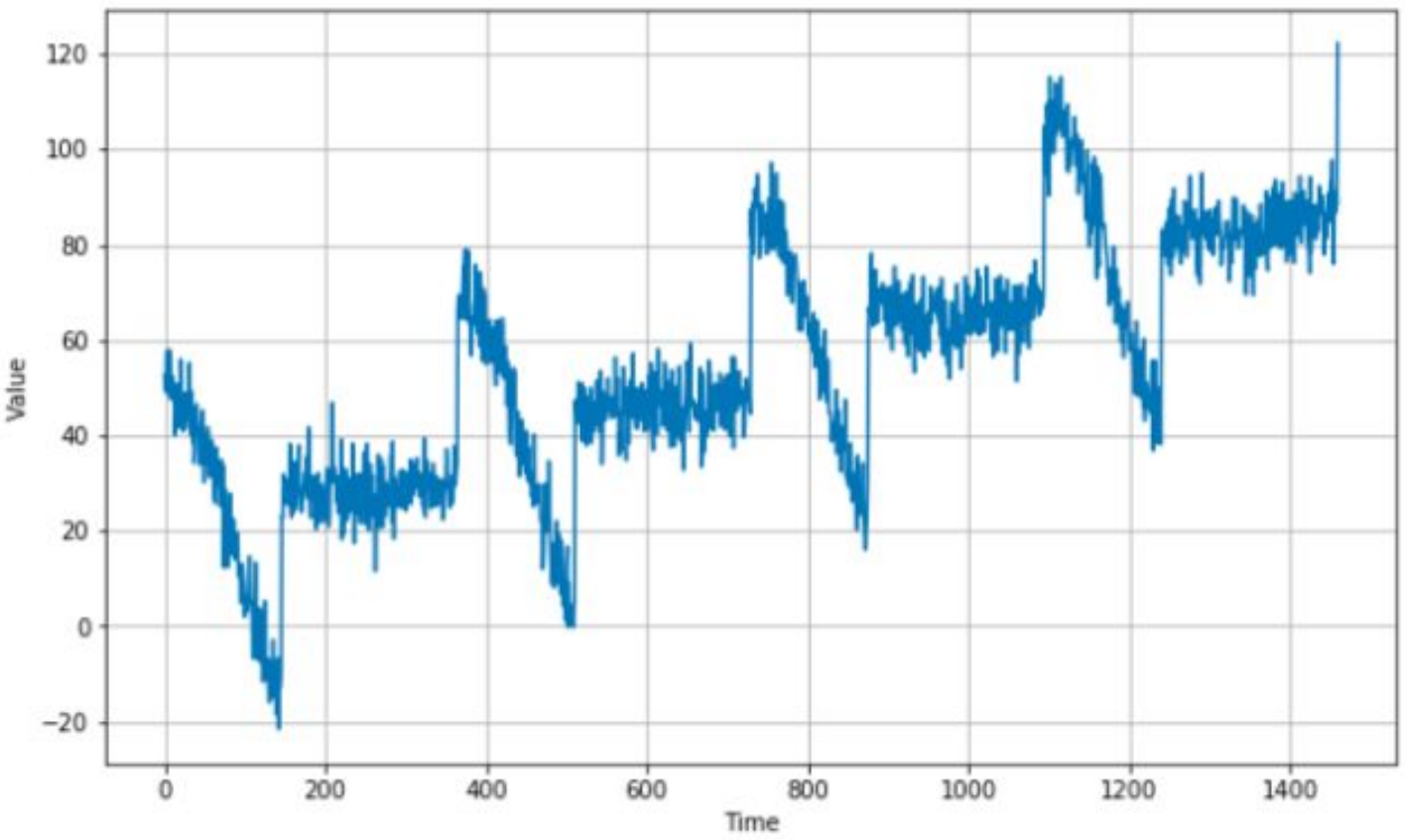
A time series whose statistical properties change over time

Non-Stationary Time Series




Non-Stationary Time Series
value
time
400
300
200
100
0
−100
0
50
100
150
200
250
300

Trend + Seasonality + Noise



TS DATA PREP

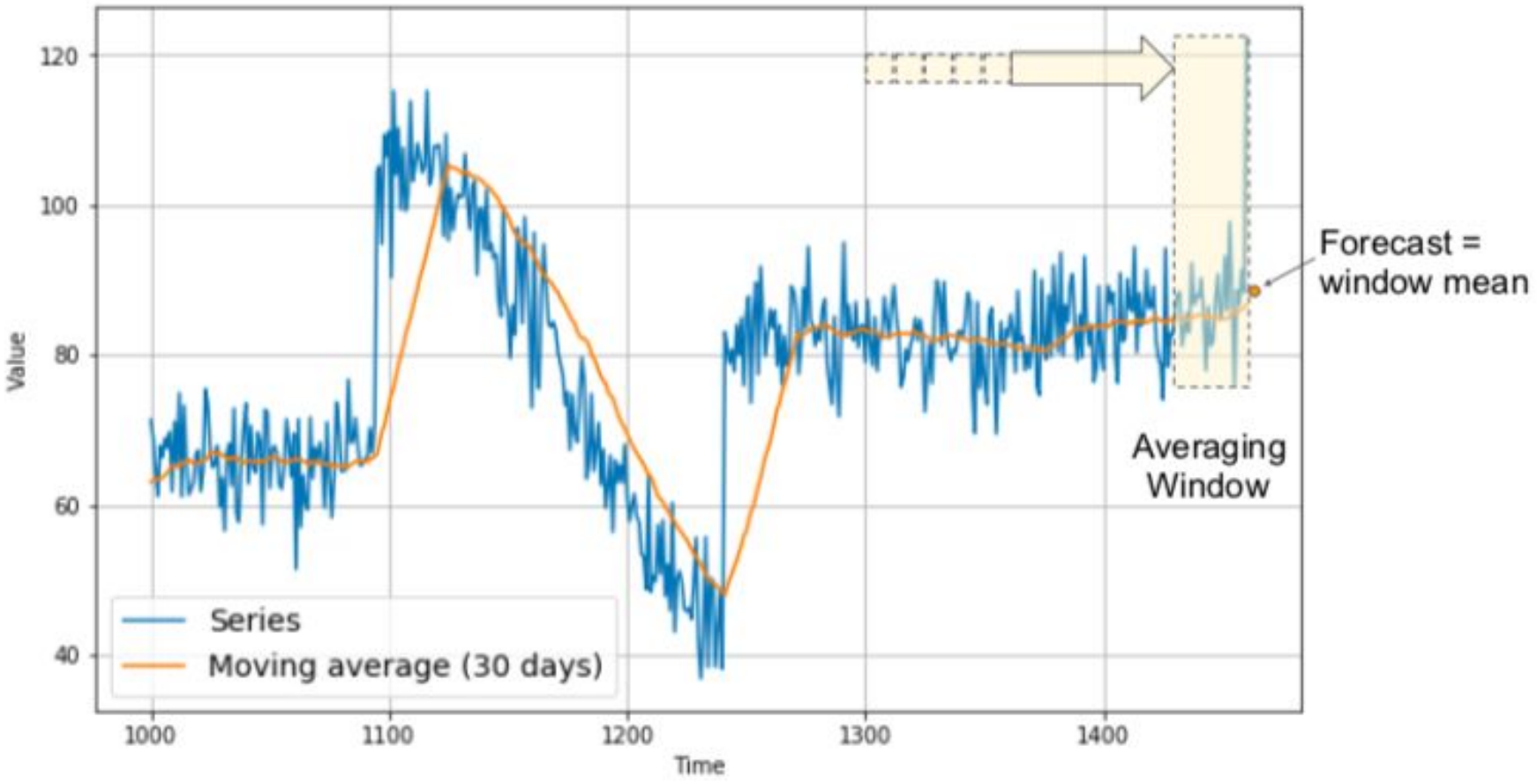
Differencing

- Time series datasets may contain trends and seasonality, which may need to be removed prior to modeling.
- Trends can result in a varying mean over time, whereas seasonality can result in a changing variance over time, both which define a time series as being non-stationary.
- Stationary datasets are those that have a stable mean and variance, and are in turn much easier to model.
- Differencing is a popular and widely used data transform for making time series data stationary.

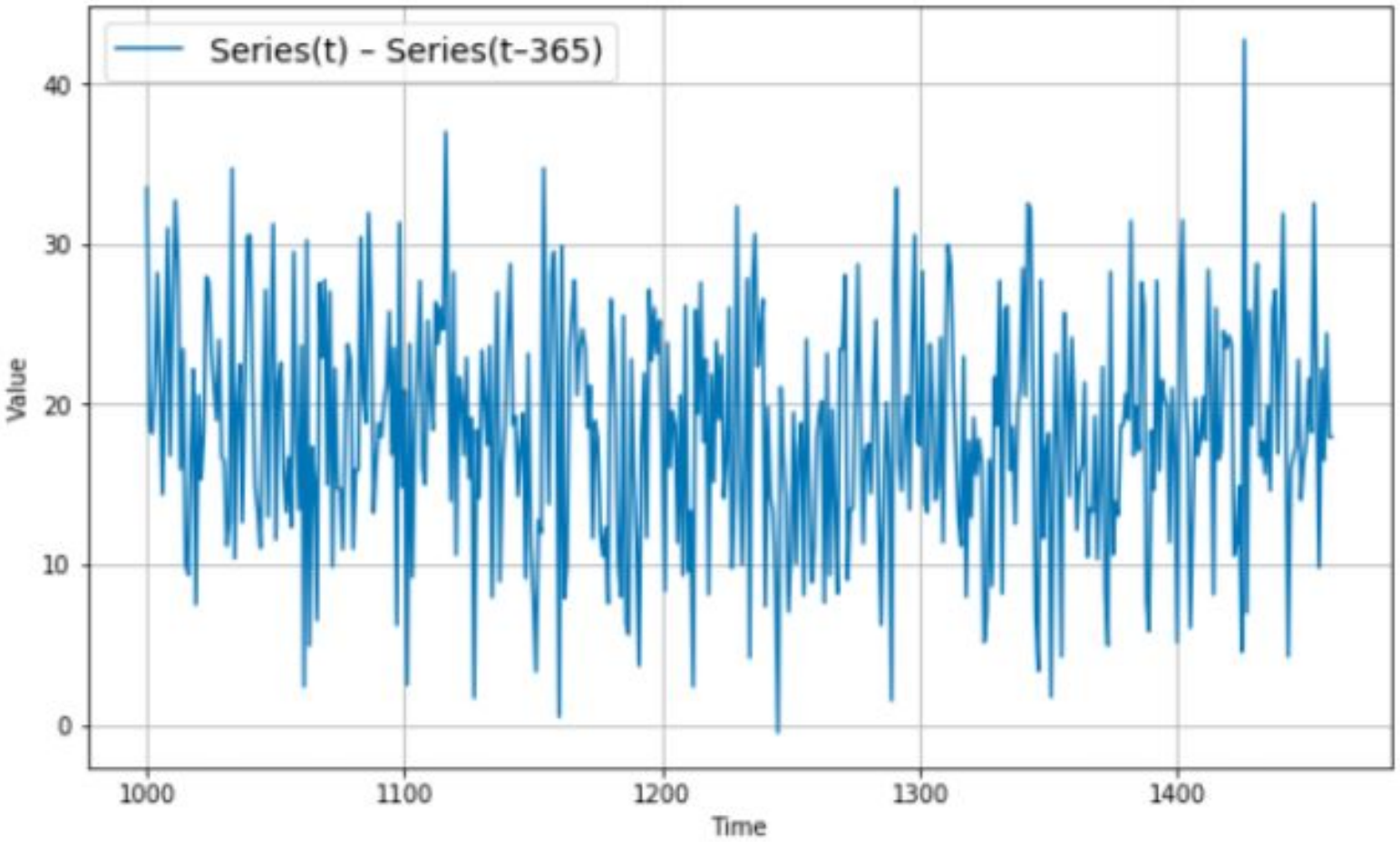
Differencing

- Differencing is a method of transforming a time series dataset.
- It can be used to remove the series dependence on time, so-called temporal dependence. This includes structures like trends and seasonality.
- Differencing is performed by subtracting the previous observation from the current observation.
  - difference(t) = observation(t) - observation(t-1)
- Inverting the process is required when a prediction must be converted back into the original scale.
- This process can be reversed by adding the observation at the prior time step to the difference value.
  - inverted(t) = differenced(t) + observation(t-1)

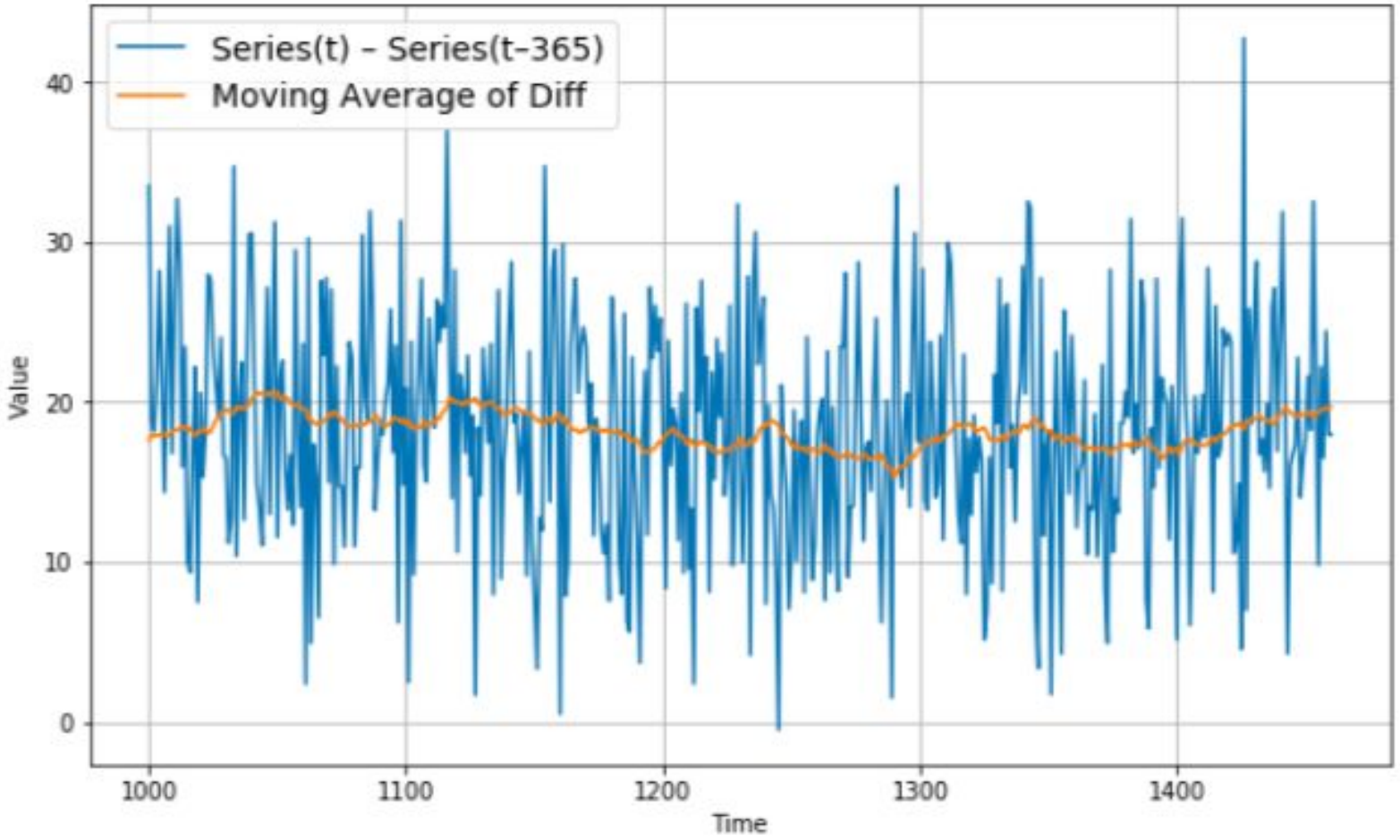
Moving Average



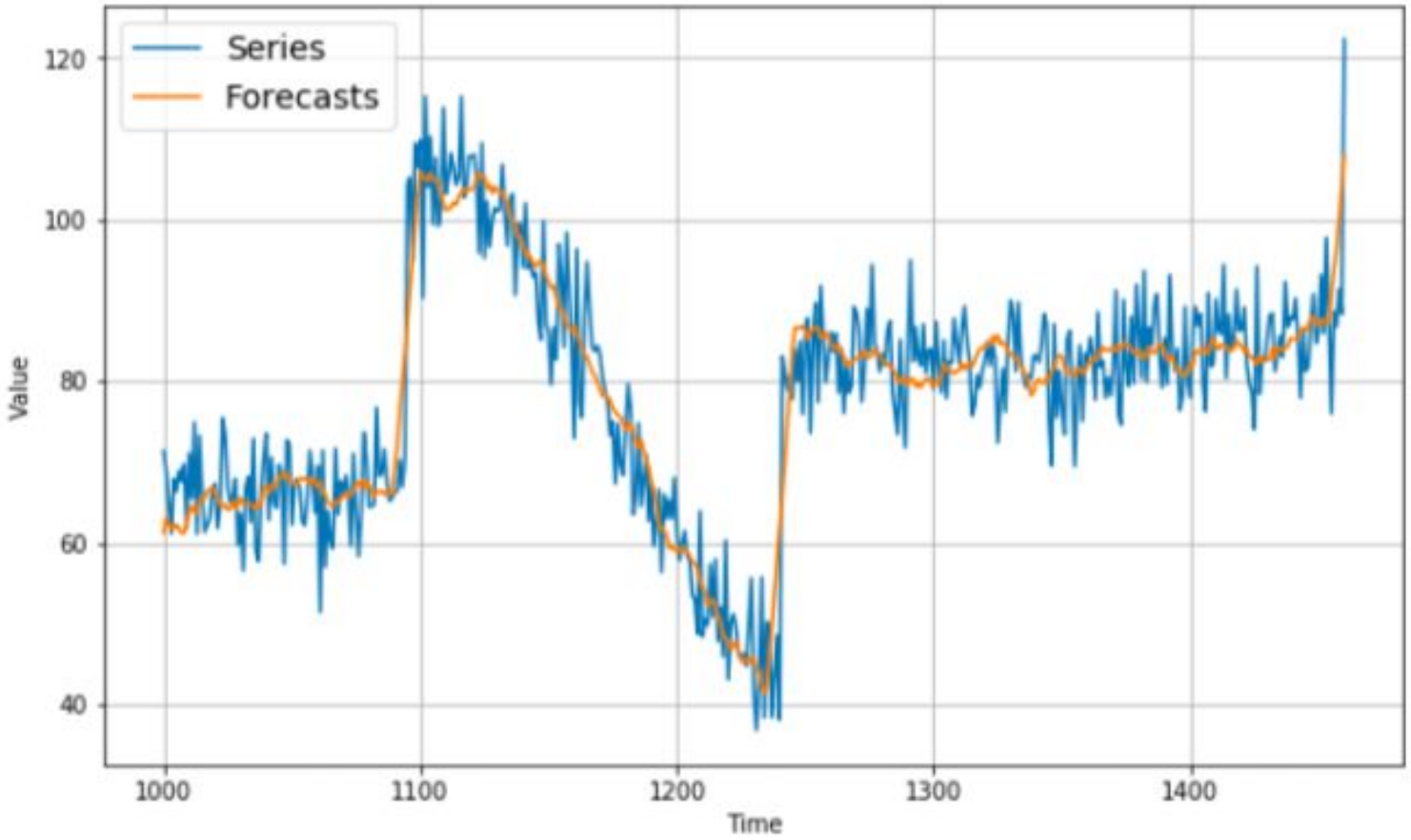
Differencing



Moving Average on Differenced Time Series

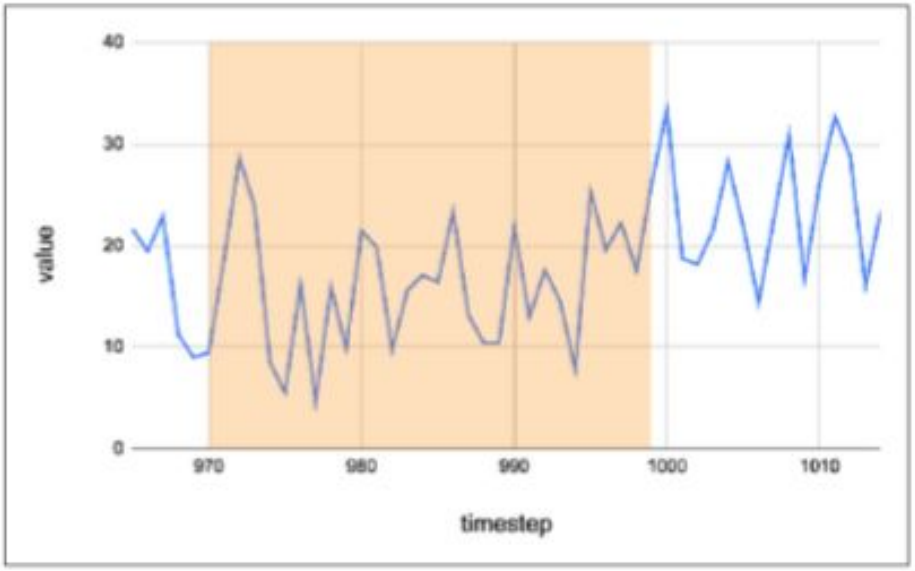


Smoothing Both Past and Present Values

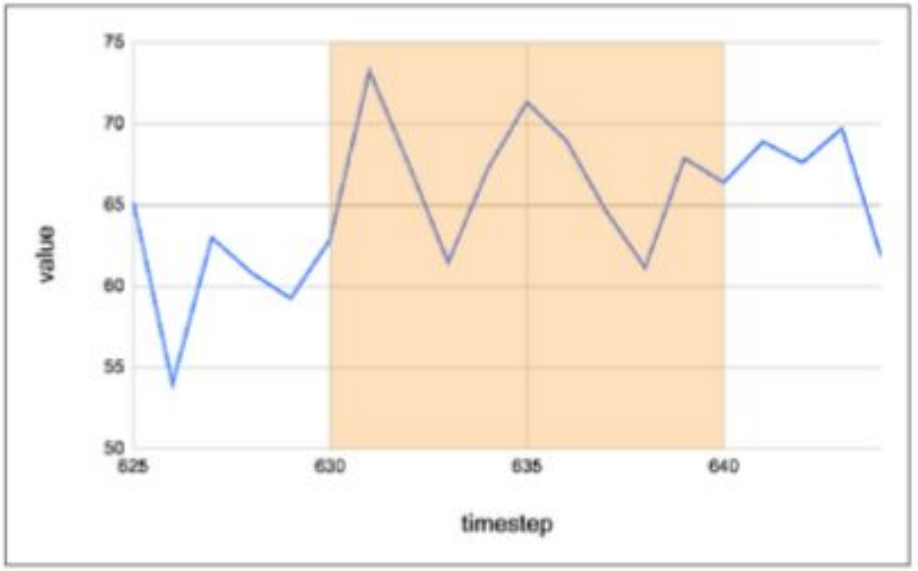


Forecasts = trailing moving average of differenced series + centered moving average of past series (t – 365)

Trailing Moving Average of Differenced Series
(zoomed at t_{1000} , window size = 30)



Centered Moving Average of Past Series (t - 365)
(zoomed at t_{635} , window size = 11)



$TMA_{t1000} = (v_{t970} + v_{t971} + v_{t972} + \ldots v_{t999}) / 30$

$CMA_{t635} = (v_{t630} + v_{t631} + v_{t632} + \ldots. v_{t640}) / 11$

forecast at $t_{1000} = TMA_{t1000} + CMA_{t635}$

Windowing

- This re-framing of your time series data allows you access to the suite of standard linear and nonlinear machine learning algorithms on your problem.
- Supervised learning is where you have input variables (X) and an output variable (y) and you use an algorithm to learn the mapping function from the input to the output.
- Time series data can be phrased as supervised learning.
- Given a sequence of numbers for a time series dataset, we can restructure the data to look like a supervised learning problem.
- We can do this by using previous time steps as input variables and use the next time step as the output variable.

Windowing

Supervised Learning Example

X, y
5, 0.9
4, 0.8
5, 1.0
3, 0.7
4, 0.9

Time Series Data

time, measure
1, 100
2, 110
3, 108
4, 115
5, 120

1-observation sliding window

X, y
?, 100
100, 110
110, 108
108, 115
115, 120
120, ?

Windowing

```
dataset = tf.data.Dataset.range(10)
for val in dataset:
    print(val.numpy())
```

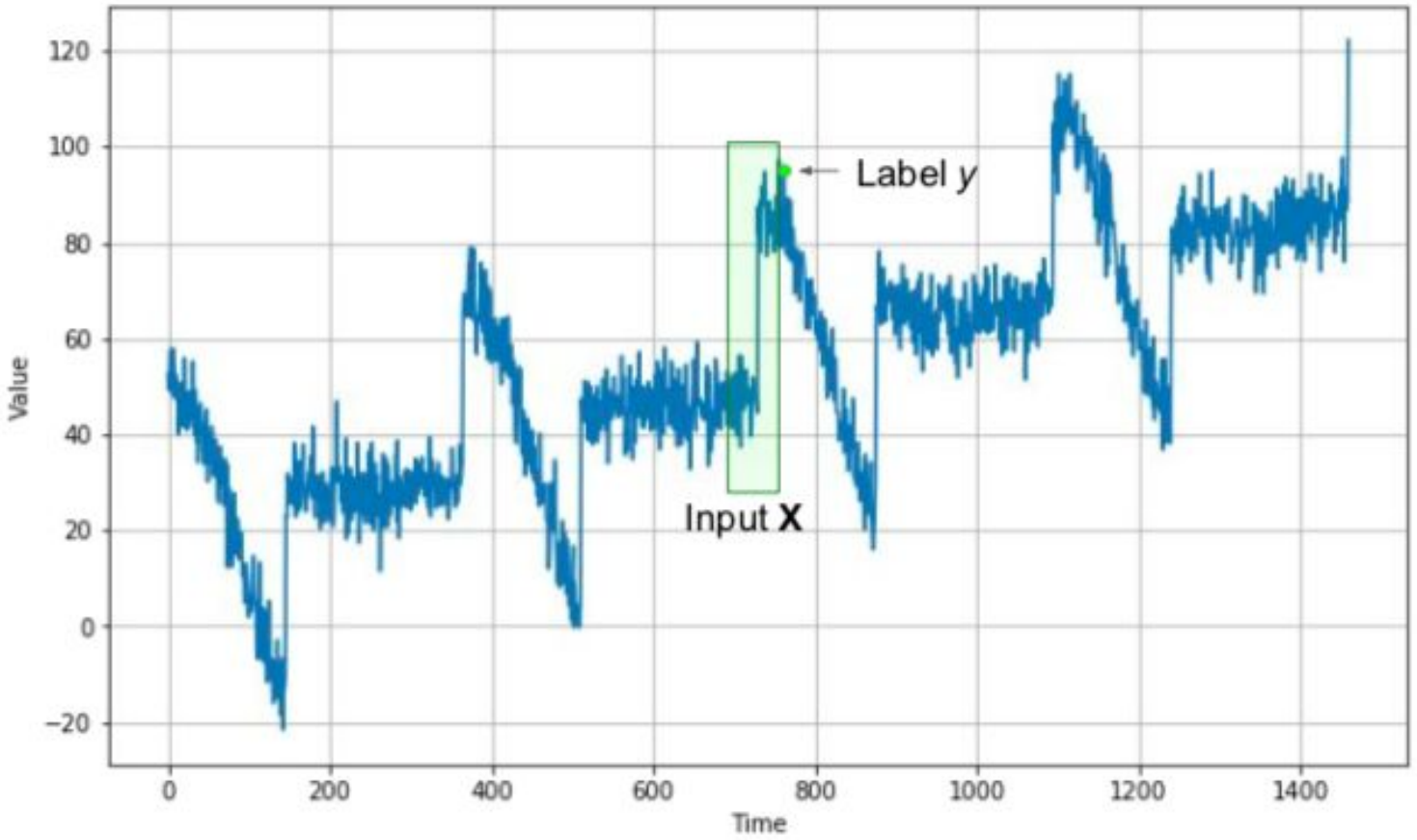
```
0
1
2
3
4
5
6
7
8
9
```

Windowing

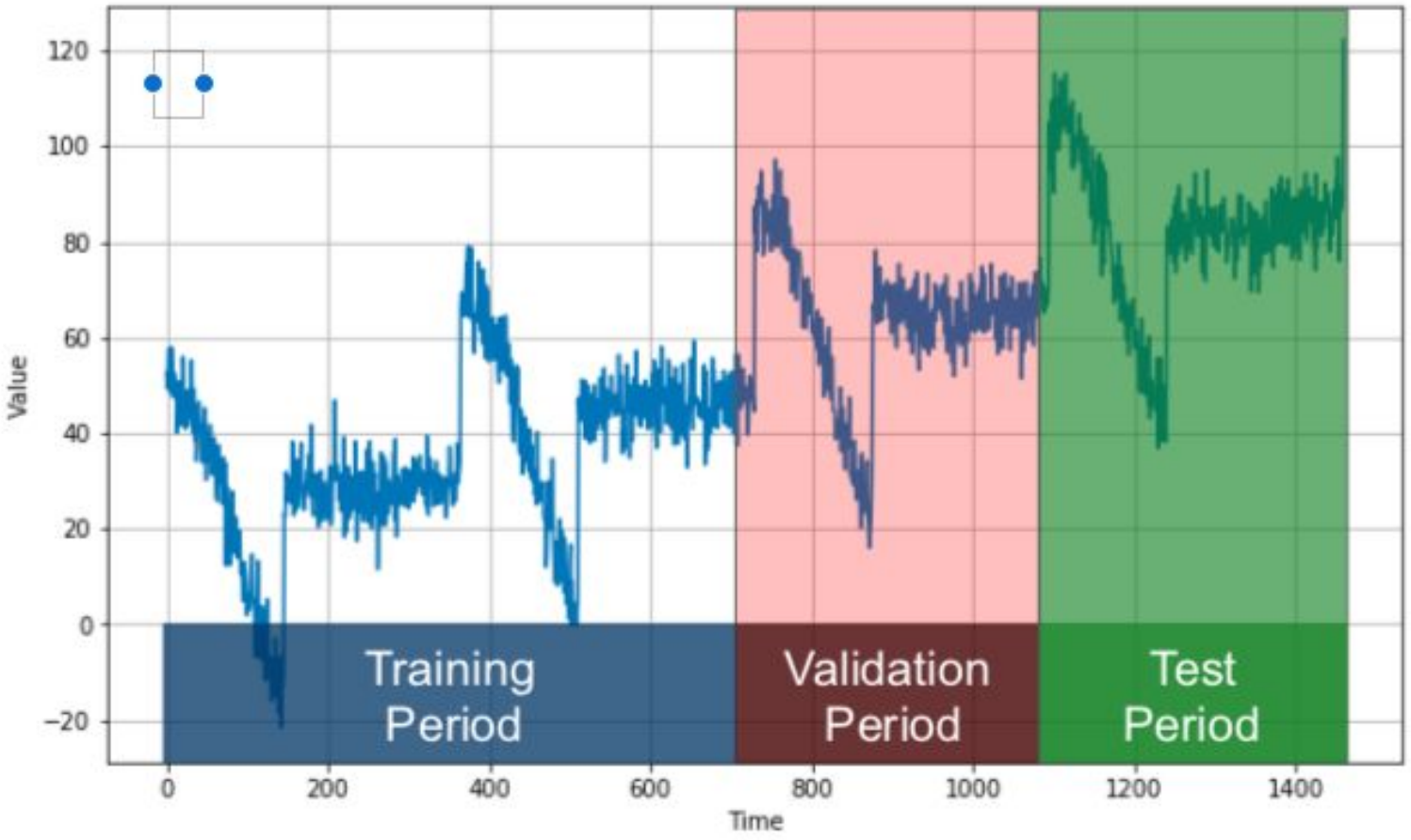
```
dataset = tf.data.Dataset.range(10)
dataset = dataset.window(5, shift=1, drop_remainder=True)
dataset = dataset.flat_map(lambda window: window.batch(5))
dataset = dataset.map(lambda window: (window[:-1], window[-1]))
for x,y in dataset:
  print(x.numpy(), y.numpy())
```

```
[0 1 2 3] [4]
[1 2 3 4] [5]
[2 3 4 5] [6]
[3 4 5 6] [7]
[4 5 6 7] [8]
[5 6 7 8] [9]
```

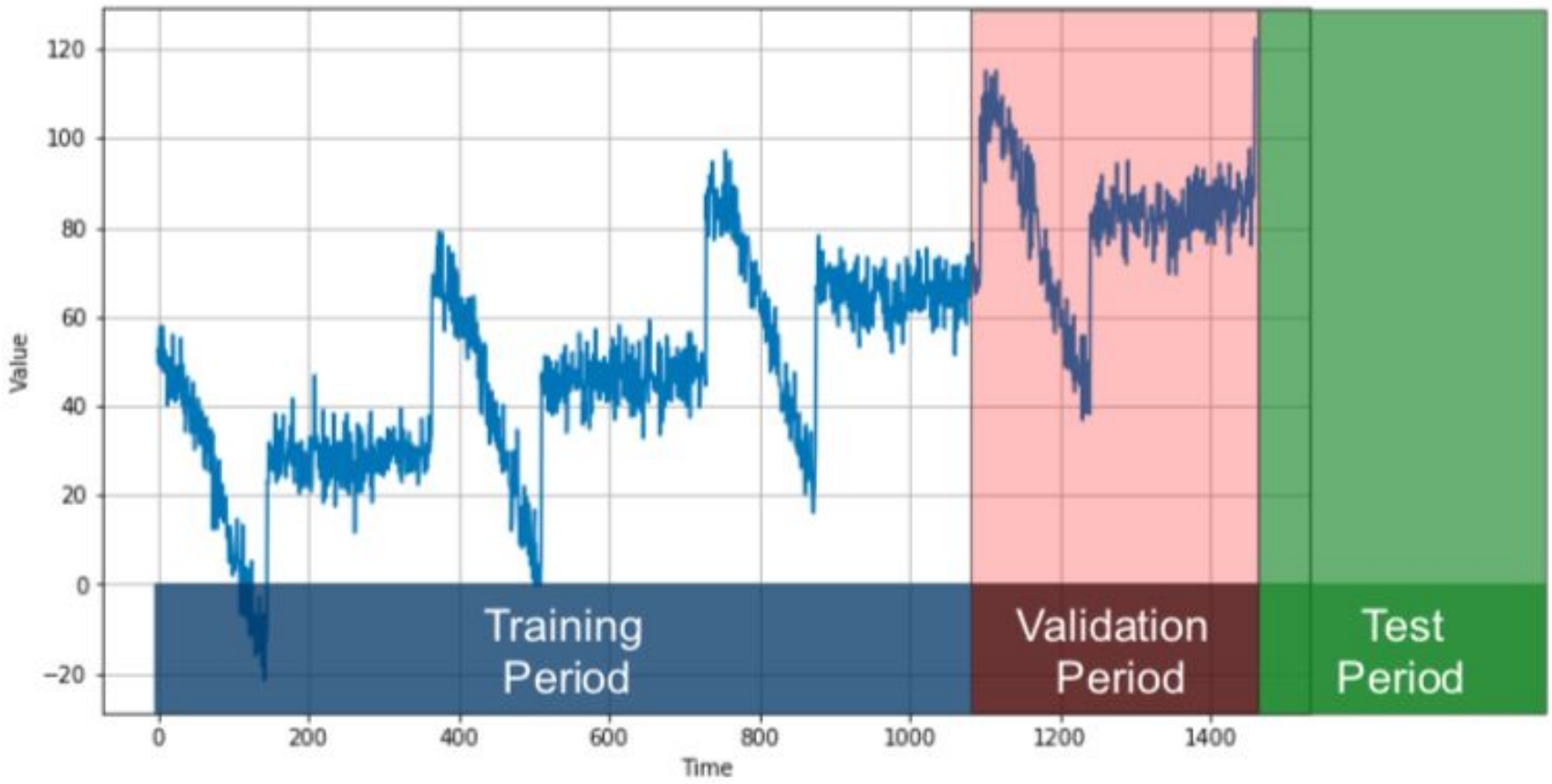
Machine Learning on Time Windows



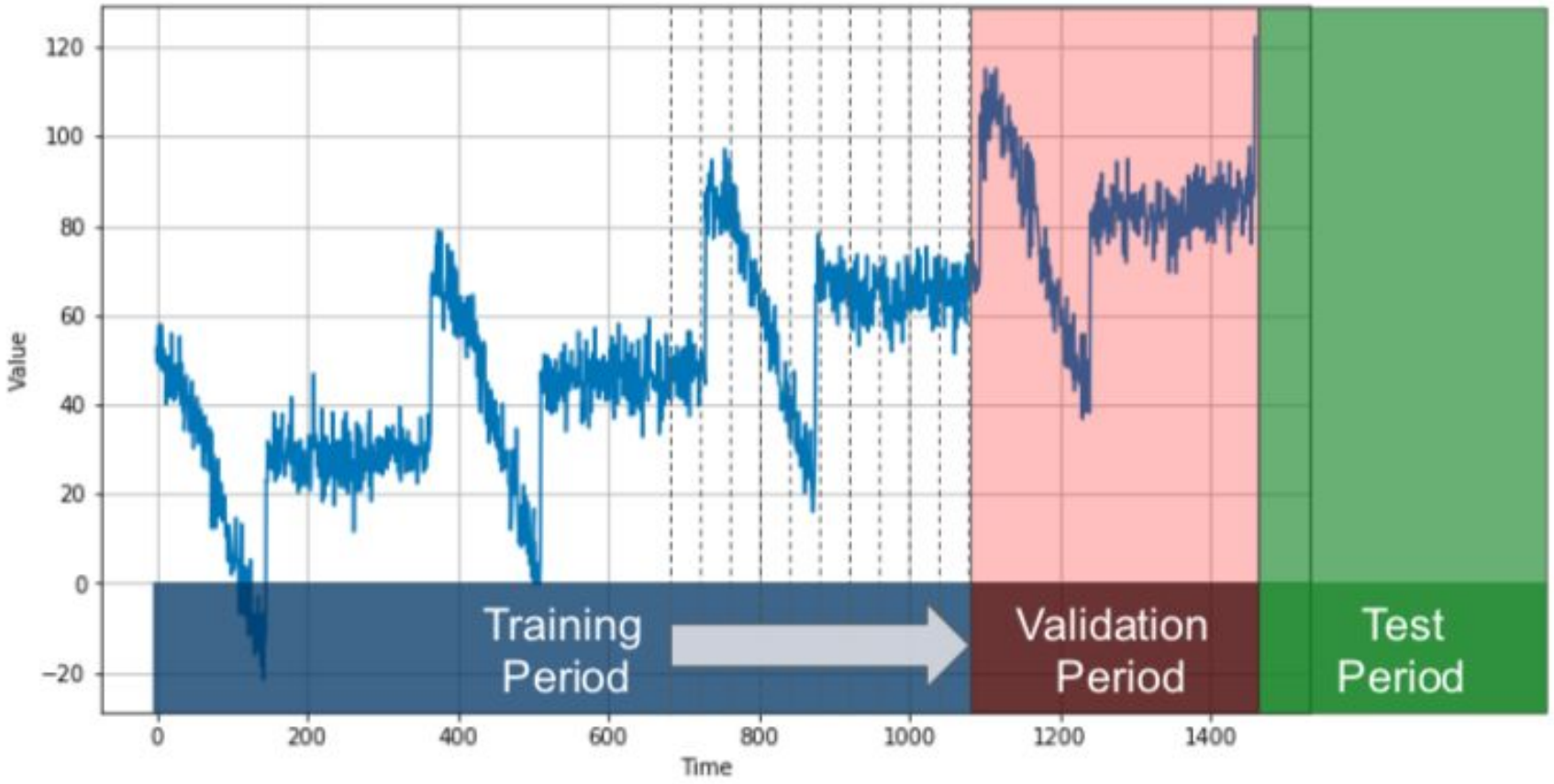
Fixed Partitioning



Fixed Partitioning



Roll-Forward Partitioning



ML models

- Lin Reg
- Simple NN
- RNN
- LSTM
- CNN

Further Reading

- https://machinelearningmastery.com/remove-trends-seasonality-difference-transform-python/
- https://machinelearningmastery.com/findings-comparing-classical-and-machine-learning-methods-for-time-series-forecasting/
- Machine Learning for Sequential Data: A Review (2002) [PDF]
- Machine Learning Strategies for Time Series Forecasting (2013) (also slides PDF)
- How to Convert a Time Series to a Supervised Learning Problem in Python