Workshop
on
Time Series Analysis in Python

Ramanathan R
Gurram Poorna Prudhvi

ODSC INDIA

Forecasting

Forecasting has been the core of many key situations in the past, present and the future (Yep - It is forecast).

Some famous forecasting quotes

> The best qualification of a prophet is to have a good memory -- Marquis of Halifax
>
> A good forecaster is not smarter than everyone else, he merely has his ignorance better organised. - Anonymous

ODSC INDIA

What is time series?

Time series is a sequence of numeric data points in successive order.

Examples:

- Prices of stocks and shares at regular intervals
- Temperature reading taken at your house at regular intervals

Important details

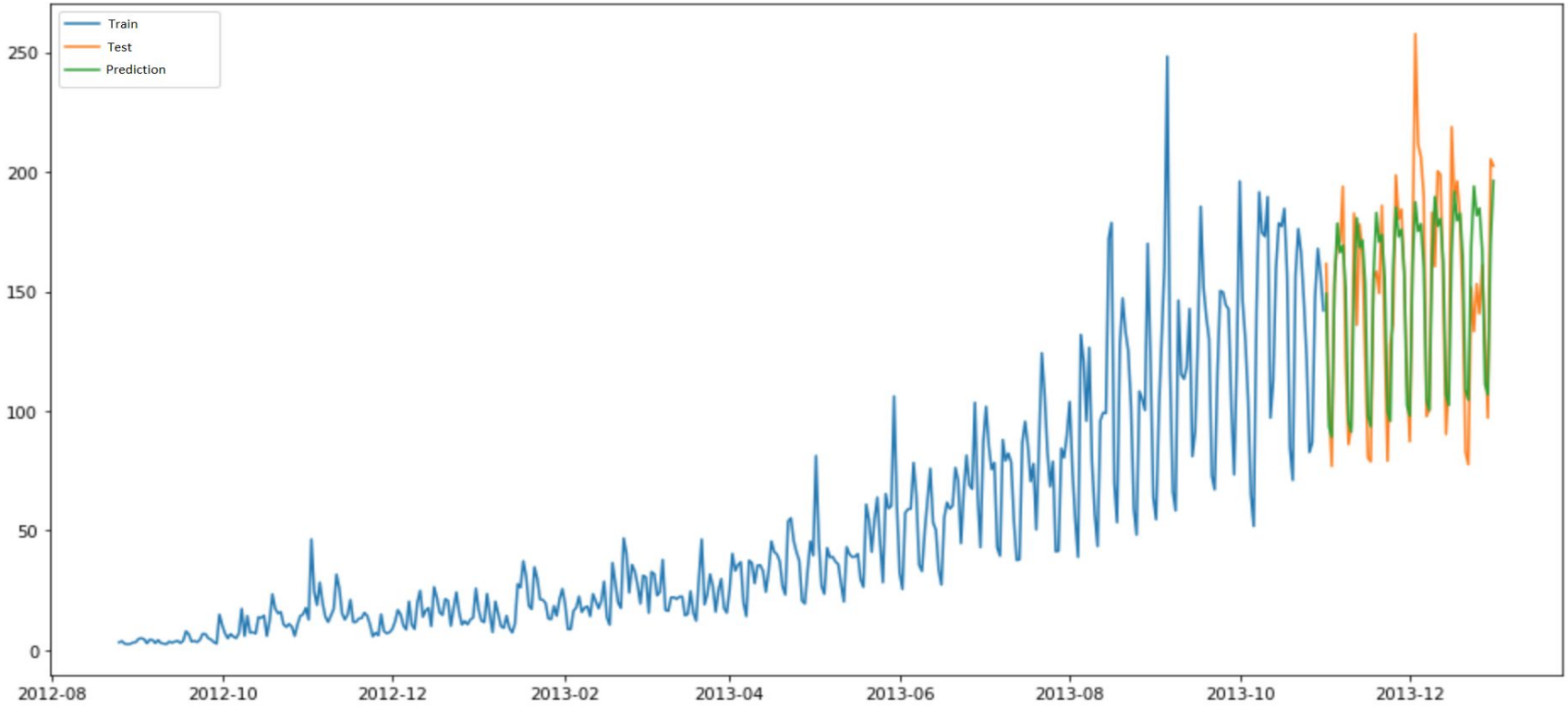
- Ordering of time point matters as it imposes certain structure to the data.
- This will help us to forecast future events with present and previous events data.

Applications

- Share price, Sales, Weather, Economics - GDP, etc,.

ODSC INDIA

Store Item Demand Forecasting Example



Workshop structure

Intro and Data Manipulation

- We start with tuning the data with pandas to make it useful for forecasting.
- Getting ourselves familiarized with time series terminology like Trend, seasonality, white noise, etc.

Building basics with statistical methods

- Focus on statistical methods like Moving Average, Auto Regression, ARMA, ARIMA, GARCH

Using Neural Networks for Forecasting :

- Focus on developing time independent neural network like MLP (Multi Layer Perceptron) and time dependent neural network LSTM (Long Short Term Memory).

Financial Time Series data :

- We will be taking a real world financial time series data and analyse it. Also, combine structured time series data with unstructured data (say news feed) and make a prediction.

Boosting:

- Focus on applying the ensemble boosting technique XGBoost to time series forecasting

ODSC INDIA

Pandas and Time Series

Statistical models

Exponential Smoothing

- Simple
- Trend
- Seasonality
- Damped Trend

Exponential smoothing - Demystified

Table 7.5: A two-way classification of exponential smoothing methods.

Trend Component	Seasonal Component		
	N	A	M
	(None)	(Additive)	(Multiplicative)
N (None)	(N,N)	(N,A)	(N,M)
A (Additive)	(A,N)	(A,A)	(A,M)
A_d (Additive damped)	(A_d,N)	(A_d,A)	(A_d,M)

Some of these methods we have already seen using other names:

Short hand	Method
(N,N)	Simple exponential smoothing
(A,N)	Holt's linear method
(A_d,N)	Additive damped trend method
(A,A)	Additive Holt-Winters' method
(A,M)	Multiplicative Holt-Winters' method
(A_d,M)	Holt-Winters' damped method

Ref: https://otexts.com/fpp2/taxonomy.html

Deep Learning For Time Series Forecasting

ODSC INDIA

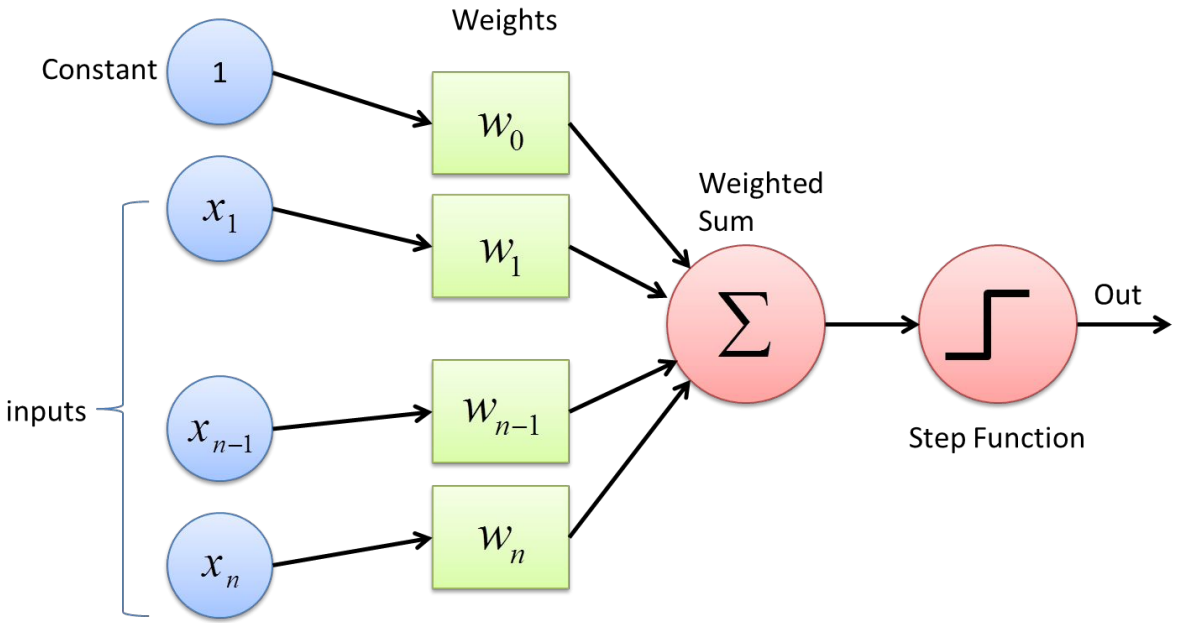
Perceptron

A perceptron produces a single output based on several real-valued inputs by forming a linear combination using its input weights (and sometimes passing the output through a nonlinear activation function). Here's how you can write that in math:

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b) = \varphi(\mathbf{w}^T \mathbf{x} + b)$$

ODSC INDIA

It looks like this

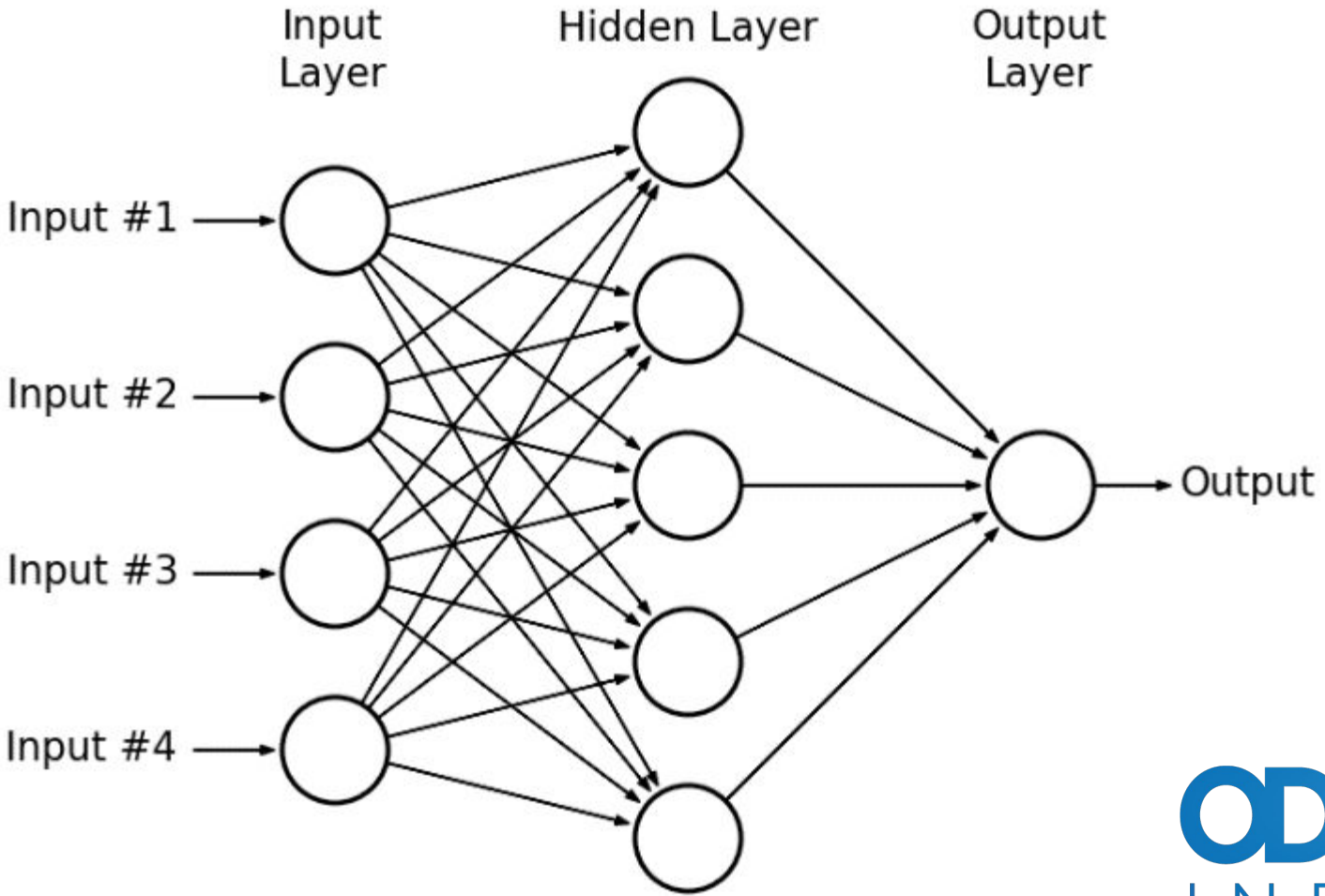


ODSC INDIA

Multi Layer Perceptron

- A multilayer perceptron (MLP) is a deep, artificial neural network. It is composed of more than one perceptron. They are composed of an input layer to receive the signal, an output layer that makes a decision or prediction about the input, and in between those two, an arbitrary number of hidden layers that are the true computational engine of the MLP.
- MLPs with one hidden layer are capable of approximating any continuous function. [universal approximation theorem which states that simple neural networks can represent a wide variety of interesting functions when given appropriate parameters]

ODSC INDIA


Input Layer
Hidden Layer
Output Layer
Input #1
Input #2
Input #3
Input #4
Output

ODSC INDIA

MLP for time series forecasting

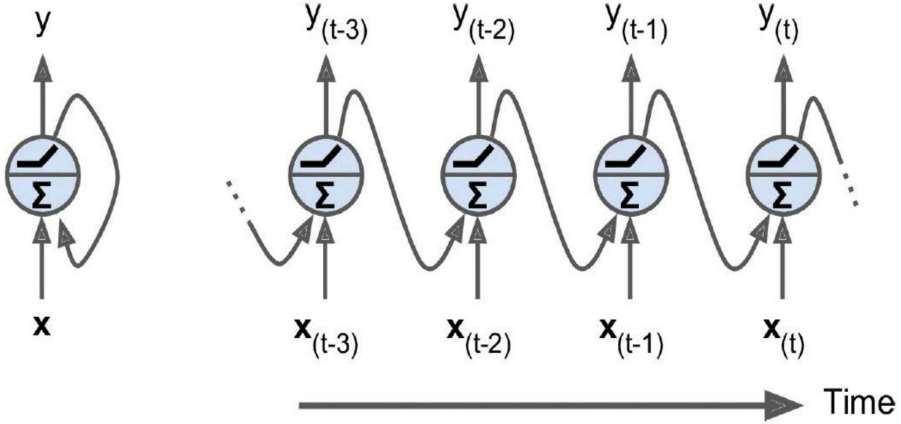
- Regression is used for forecasting
- For example we are modeling the time series using day wise data and if we use past 7 days as the features for forecasting this becomes equivalent to AR(7) as seen before.
- MLP doesn’t capture the temporal patterns

ODSC INDIA

Let’s check out how to model time series data using Multi Layer Perceptron

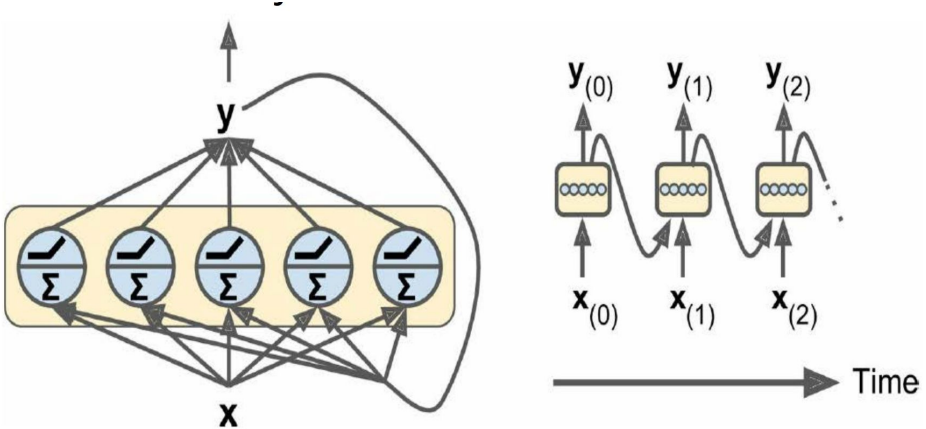
Recurrent Neurons

A recurrent neuron (RN, the simplest possible RNN) on the left is unrolled through time on the right. RN looks like feedforward neuron, except it has connections pointing backward. At each time step t, RN just has one neuron receiving inputs, producing an output, and sending that output back to itself.



ODSC INDIA

A Layer of Recurrent Neurons

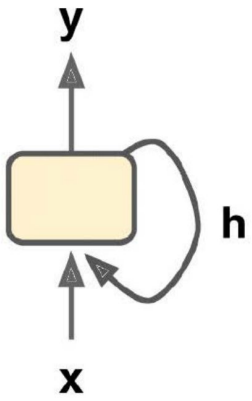


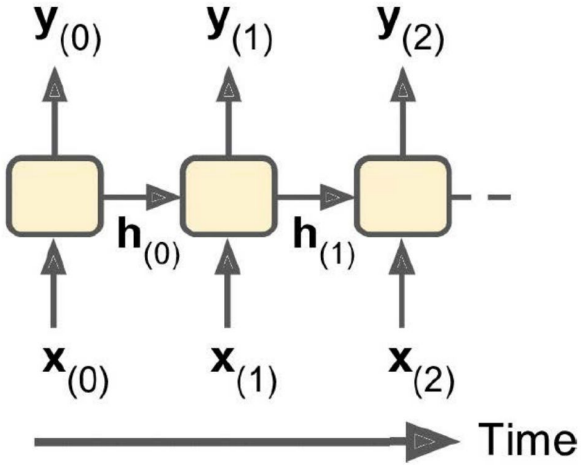
A layer of 5 RNs as a cell, both the inputs and outputs are vectors Each RN has 2 sets of weights: 1 for the inputs x(t) and the other for the outputs of the previous time step y(t–1) y(t) is a function of x(t) and y(t–1), which is a function of x(t–1) and y(t–2), and so on. So, y(t) a function of all the inputs since t = 0 (x(0),x(1), …, x(t)), and y(-1) is assumed to be all 0.

ODSC INDIA

Memory Cell

- Since y(t) a function of all the inputs since t = 0, the part of RN preserving states across time axis is a “Memory Cell” (“Cell”)
- A cell’s state at t is h(t)=f(h(t-1), x(t)), where x(t) is current inputs, h(t-1) is cell’s state at t-1.
- For single RN or a layer of RNs, y(t)=h(t). But for complex cells, y(t)!=h(t), i.e. the LSTM cell





ODSC INDIA

Problems with RNN

Suffer from the vanishing/exploding gradients (train forever) :

- Tricks: parameter initialization, ReLU activation function, batch normalization, gradient clipping, faster optimizer
- Training still be very slow for a moderate long sequence (i.e. 50 steps)
- Truncated backpropagation through time: unroll the RNN only over a limited number of time steps during training by simply truncating the input sequences
- Model will not be able to learn long-term patterns.

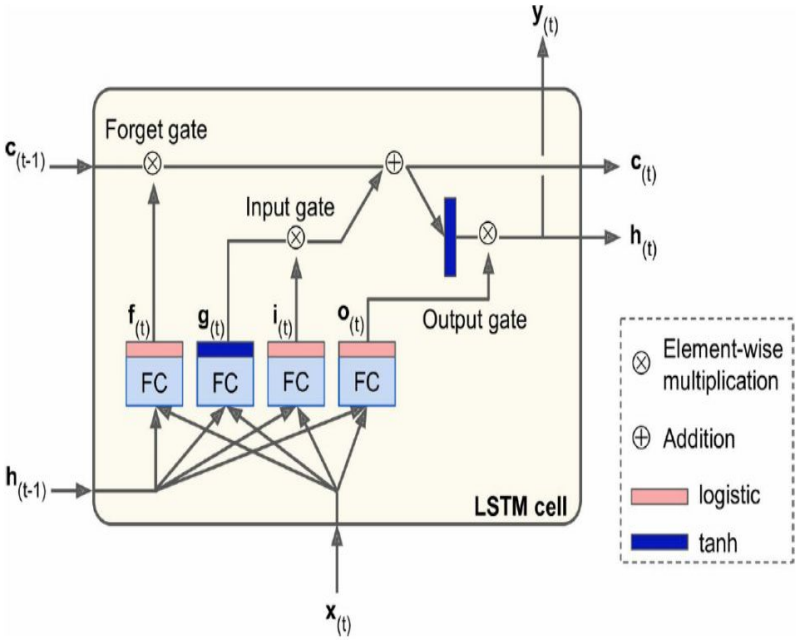
ODSC INDIA

Memory of those first inputs gradually fades away

- Some information is lost after each time step, due to the transformation of data when it traverse an RNN.
- After a few time steps, RNN’s state hardly contains the information from those first inputs.

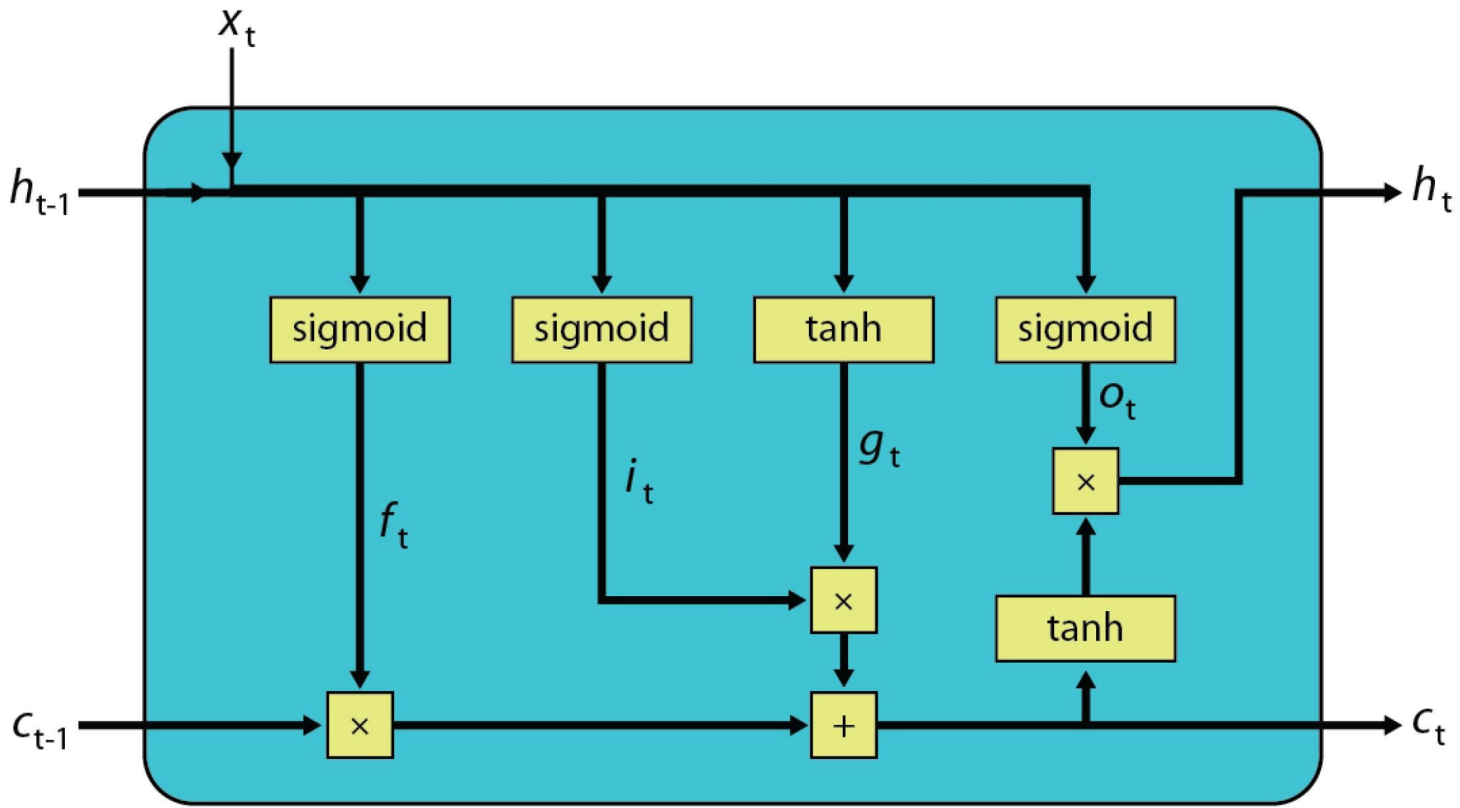
Basic LSTM Cell

- Its training will converge faster and detect long-term dependencies in the data
- Its state is split in two vectors: h(t) as short-term state, c(t) as long-term state.
- c(t-1) first drop some previous memories, then add some new current memory
- After addition, c(t-1) is copied and passed through “tanh” and “Output gate” o(t)



ODSC INDIA

An view of gates in LSTM Cell



Let’s check how to model time series forecasting using LSTM

Q & A