MIS780 – Advanced AI for Business

Week 6

Deep Learning III
(Recurrent Neural Nets)

Dr Anagi Gamachchi

Discipline of Information Systems and Business Analytics,
Deakin Business School

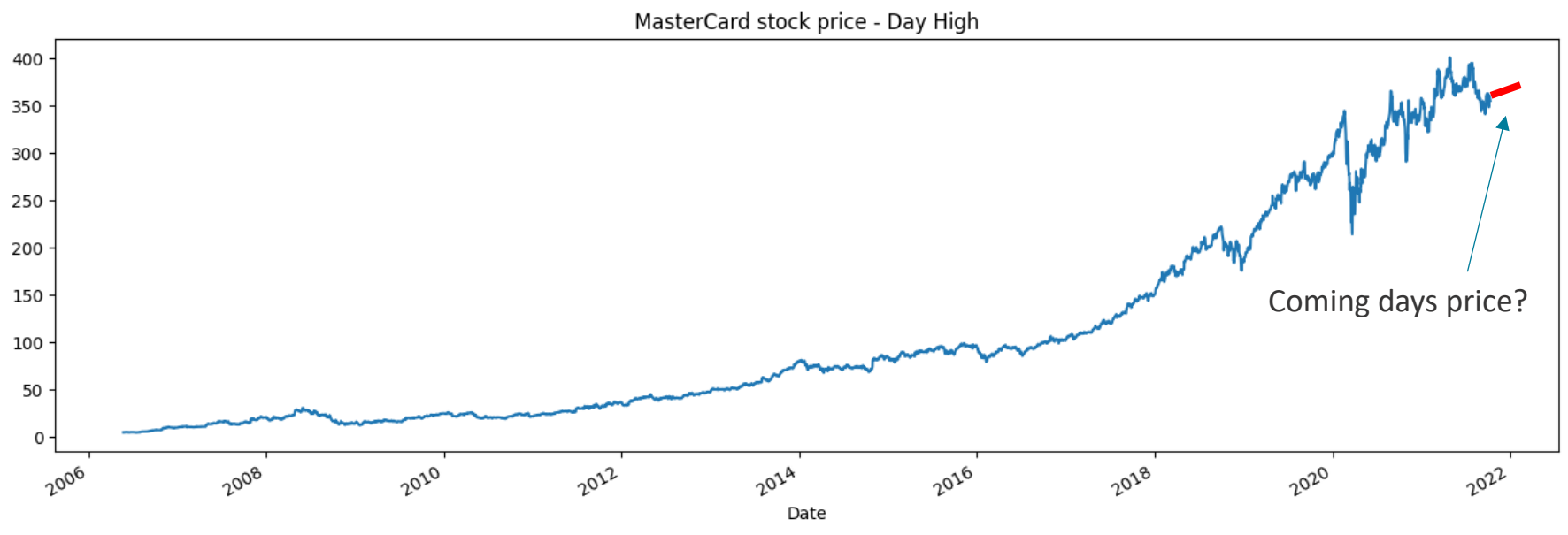
Time Series Forecasting

Data Set:

MasterCard Stock Data (source: Kaggle)

Daily price of stock price and volume from 2006 to 2021

Date	Open	High	Low	Close	Volume
25/05/2006	3.7489667	4.2838687	3.7396638	4.2792172	395343000
26/05/2006	4.3071262	4.3480576	4.1033979	4.1796799	103044000
30/05/2006	4.1834002	4.1843304	3.9861838	4.0931644	49898000
31/05/2006	4.1257227	4.2196792	4.1257227	4.1806083	30002000
1/06/2006	4.1796782	4.4745719	4.1768872	4.4196863	62344000
2/06/2006	4.5117821	4.5303874	4.3527069	4.3713121	37253000
5/06/2006	4.3768952	4.5815537	4.3722438	4.5722508	37188000



Problem: Forecast the **High** price of MasterCard Stock in the coming days (7 days).

Solution: Build a time-series forecasting model to predict stock price in the coming 7 days

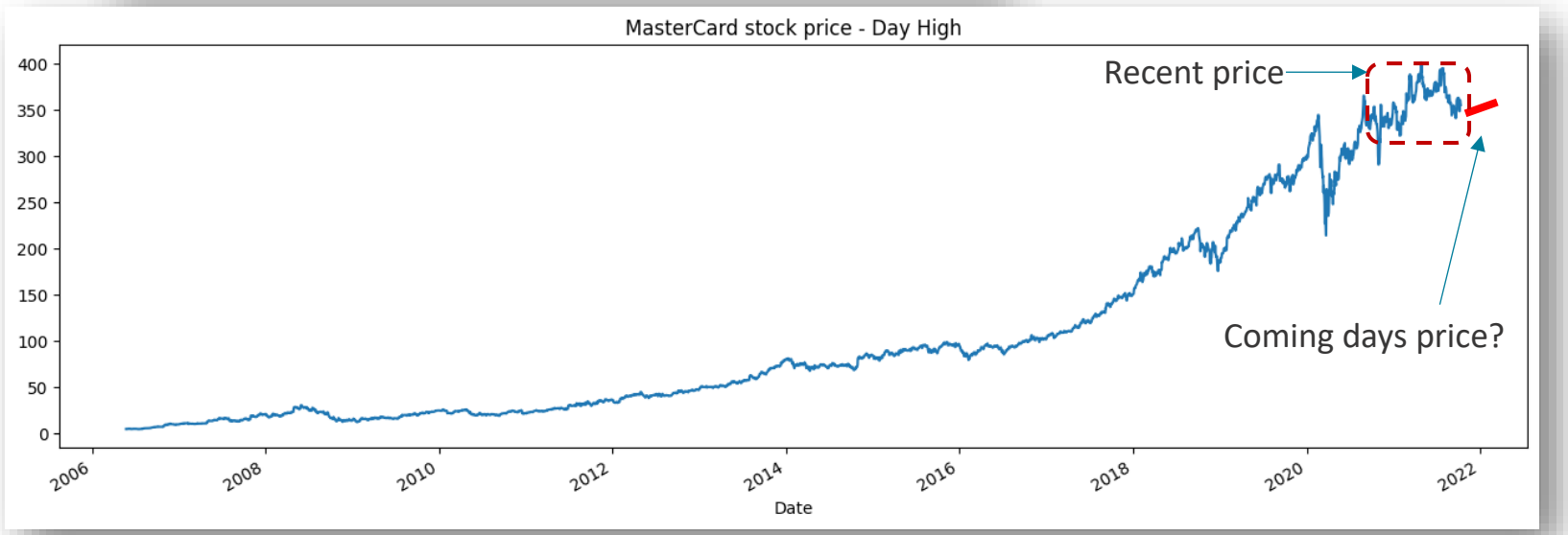
Source: https://medium.com/@fenjiro/time-series-for-business-a-general-introduction-50968346e660

Uni-Variate Time Series Forecasting

Date	Open	High	Low	Close	Volume
25/05/2006	3.7489667	4.2838687	3.7396638	4.2792172	395343000
26/05/2006	4.3071262	4.3480576	4.1033979	4.1796799	103044000
30/05/2006	4.1834002	4.1843304	3.9861838	4.0931644	49898000
31/05/2006	4.1257227	4.2196792	4.1257227	4.1806083	30002000
1/06/2006	4.1796782	4.4745719	4.1768872	4.4196863	62344000
2/06/2006	4.5117821	4.5303874	4.3527069	4.3713121	37253000
5/06/2006	4.3768952	4.5815537	4.3722438	4.5722508	37188000

Uni-variate Forecasting:
Future is predicted by on historical data of a single time series

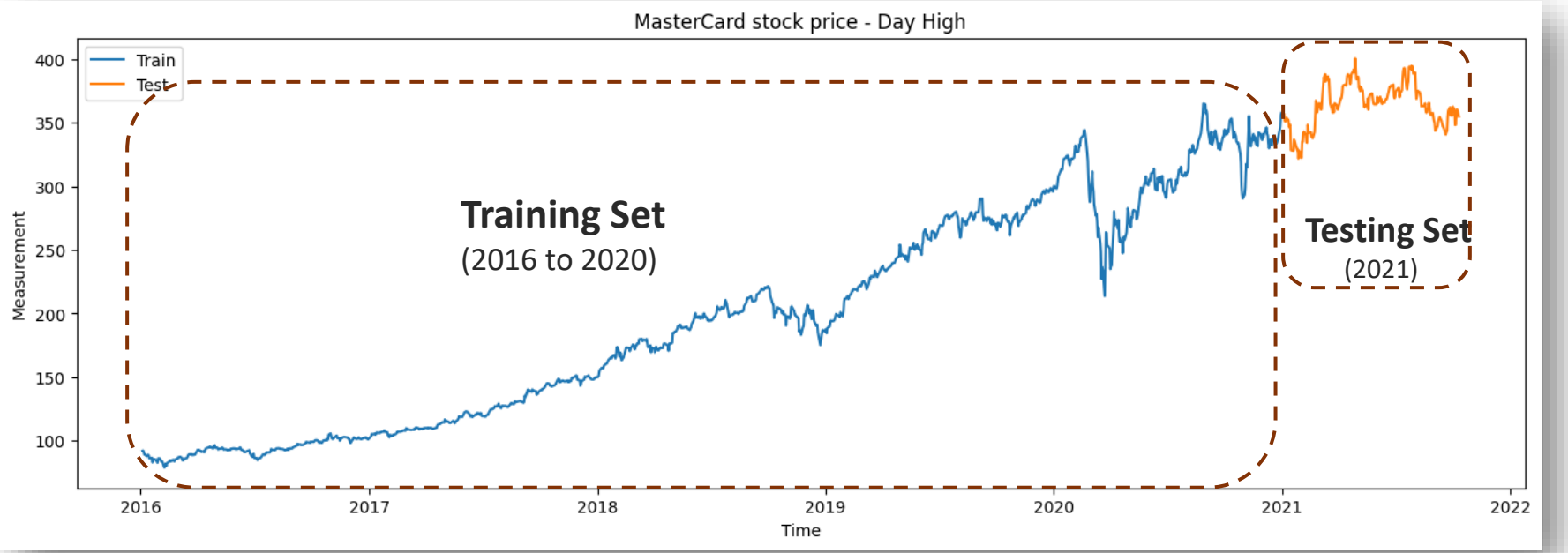
Forecasting Problem:
Predict the **High** price of Mastercard Stock of the coming 7 days based on historical data of daily **High** price alone.



Assumption:
The High price of coming days are depending on **recent** High price (e.g., recent few weeks)

How to construct training data to train the forecasting model?

Data Preparation

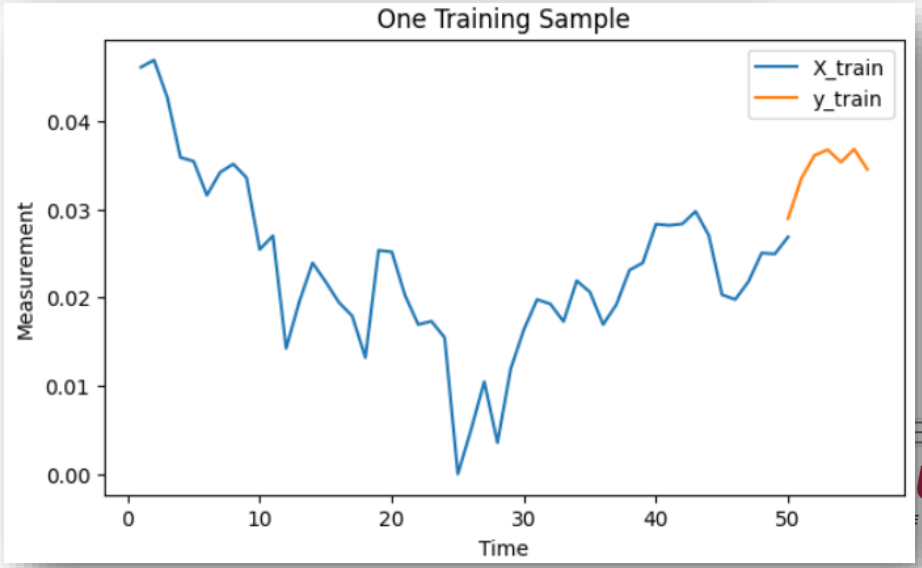


Historical data (prior to 2016) are considered old (useless for prediction), and thus not used.

Training and Testing Data Construction

Assume that the sequence is [1,2,3,4,5,6,7,8,9,10,11,12] and the **n_step** is four (*representing historical data used in forecasting*) and the **forecasting_horizon** is two (*representing the number of forecasting step ahead*)

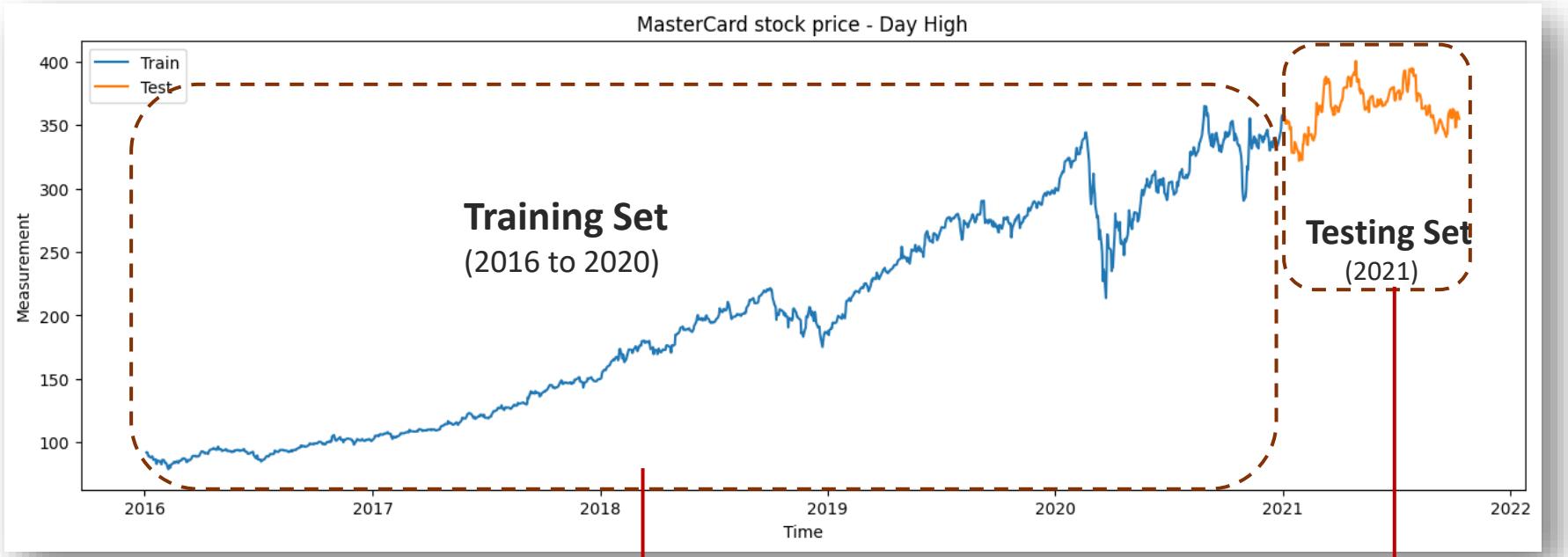
x	y
1, 2, 3, 4	5, 6
2, 3, 4, 5	6, 7
3, 4, 5, 6	7, 8
4, 5, 6, 7	8, 9
5, 6, 7, 8	9, 10
6, 7, 8, 9	10, 11
7, 8, 9, 10	11, 12



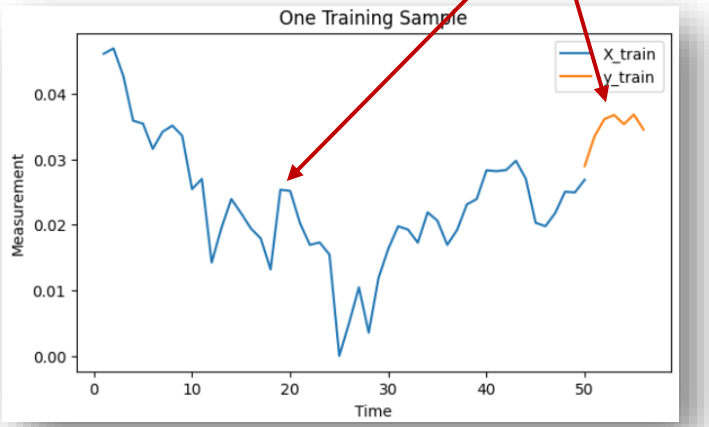
Our experiment

n_step = 50
forecasting_horizon = 7

Data Preparation




```
X_train shape: (1203, 50, 1)
y_train shape: (1203, 7, 1)
```



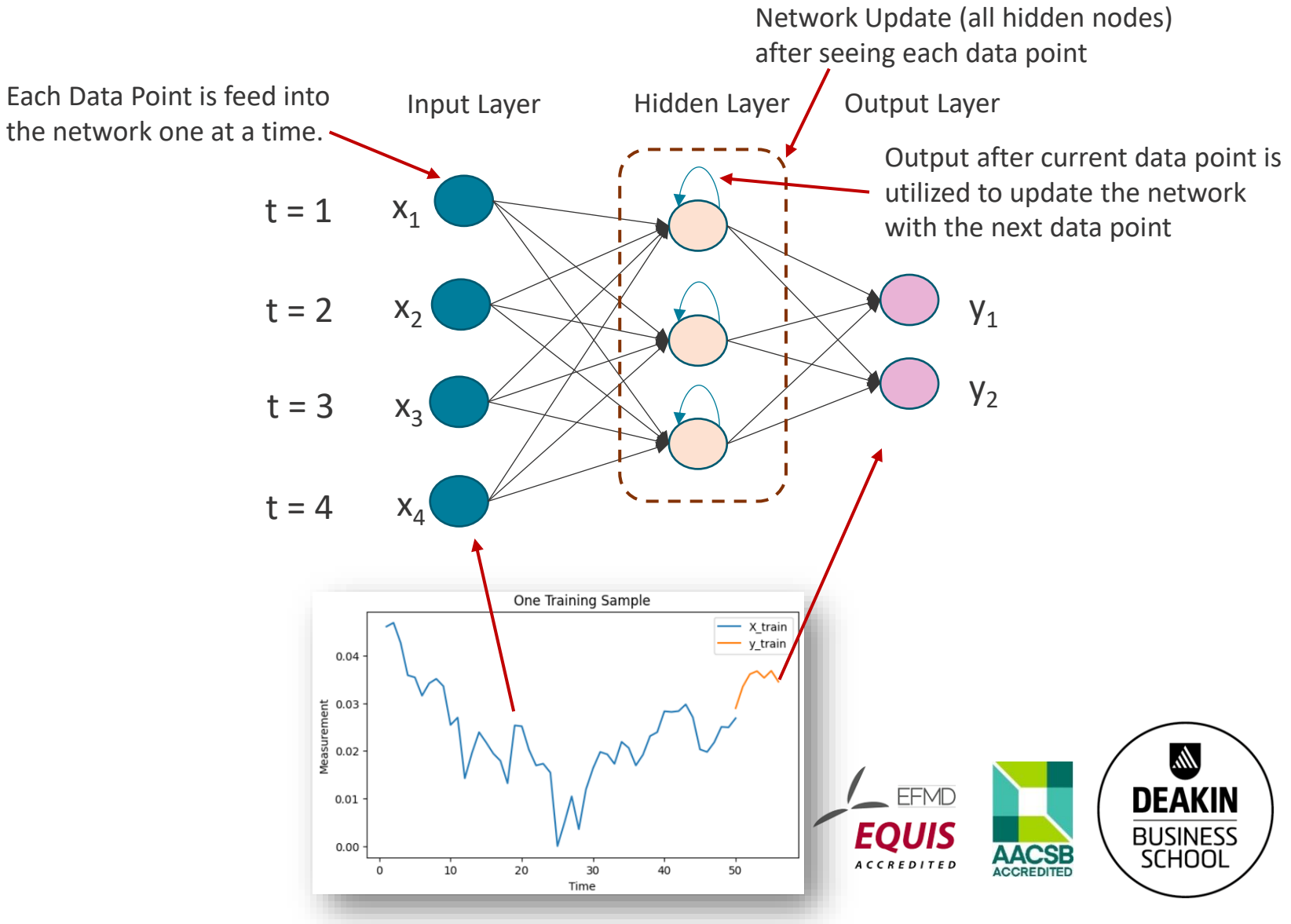

```
X_test shape: (139, 50, 1)
y_test shape: (139, 7, 1)
```

Recurrent Neural Networks

- ***RNN*** is a generalization of feedforward neural network that has an internal memory, and connections between nodes form a sequence.
- In traditional neural networks, all the inputs are independent of each other.
- In **RNN**, all the <u>inputs are related</u> to each other.

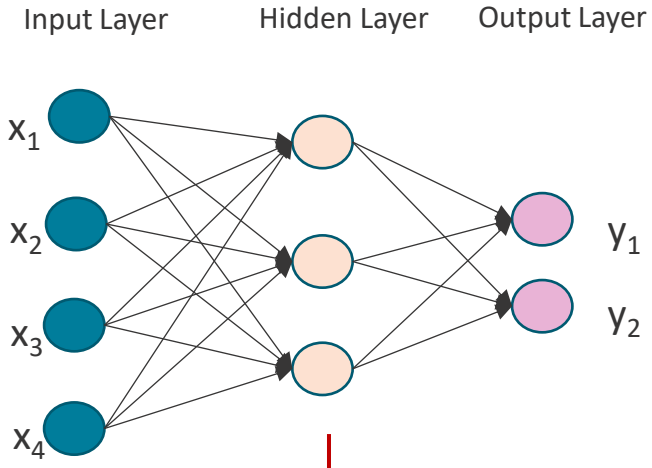
RNN Architecture for Time Series Forecasting

(e.g., forecasting stock prices for next **2 days** based on last the **4 days price**)



Hidden Node Architecture

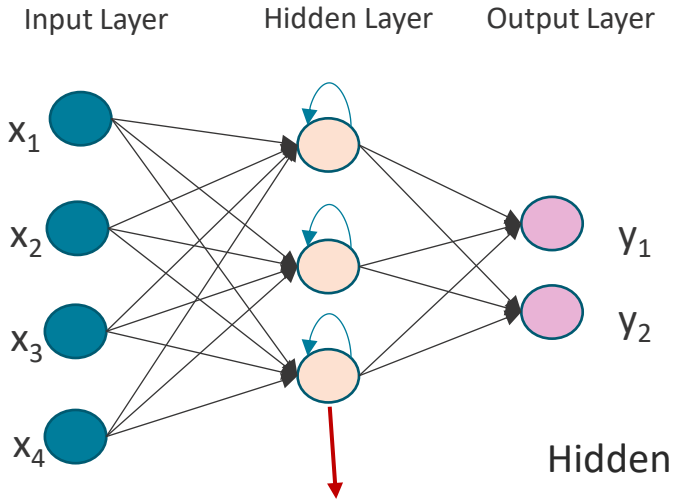
Traditional Neural Networks



One Hidden Node (ANN)

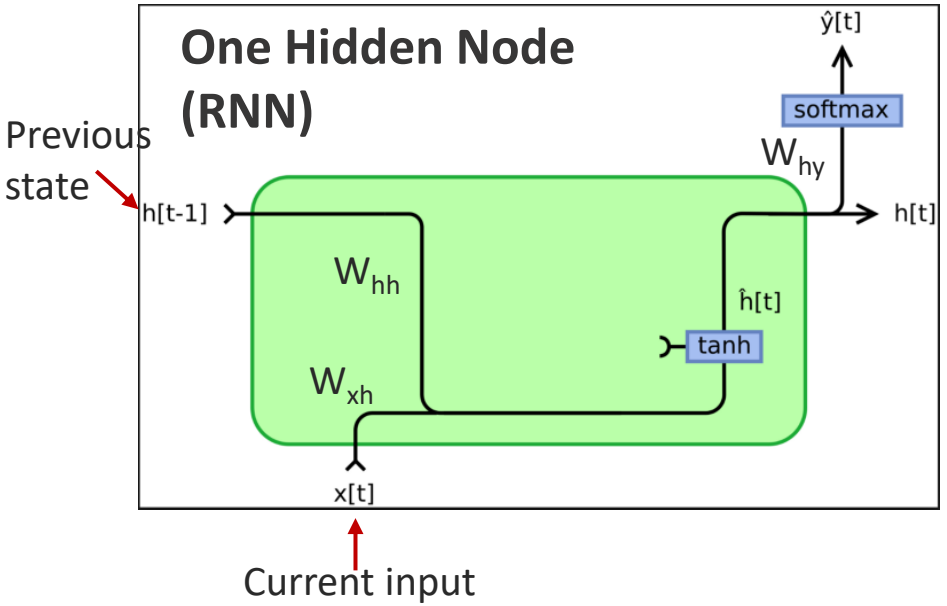
$$W_0 + W_1 \times X_1 + \ldots + W_m \times X_m$$

Recurrent Neural Networks



Hidden Node Output 2

$$y_t = W_{hy} h_t$$

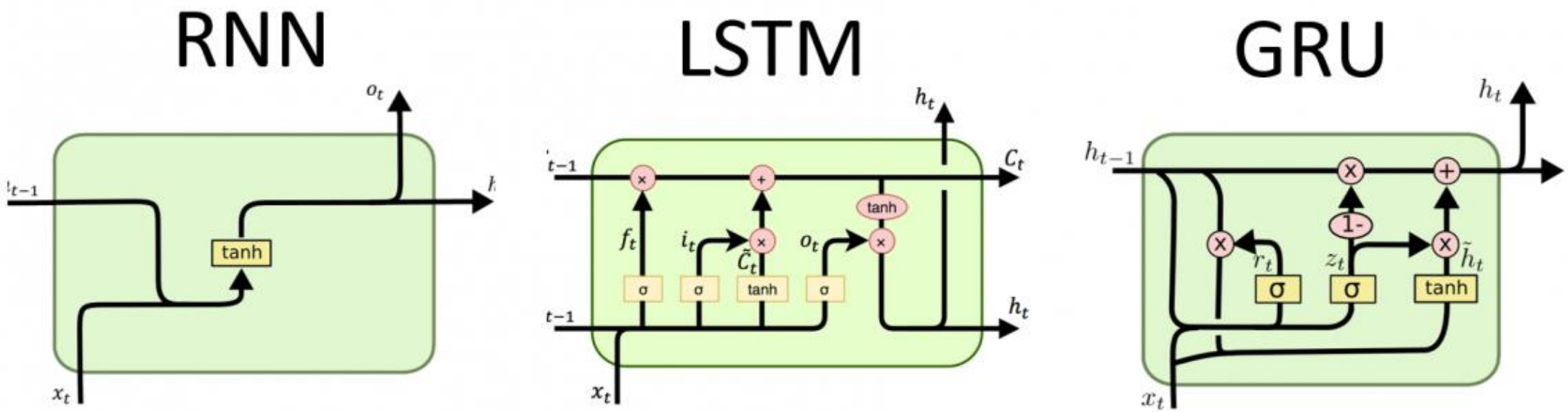


Hidden Node Output 1

$$h_t = \tanh (W_{hh} h_{t-1} + W_{xh} x_t)$$

Recurrent Neural Networks (cont.)

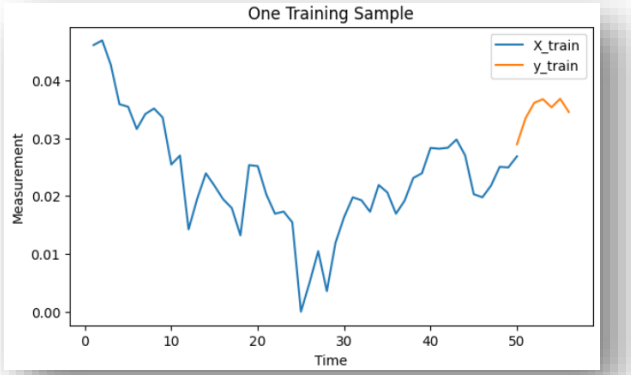
- The hidden node for RNN shown above suffers from the inability to retain information when the sequence given is long. It forgets information that was supplied several time-steps ago, which limits the learning performance.
- So people created some upgraded architectures to tackle that. The most popular are **Long short-term memory (LSTM)** and **Gated Recurrent Unit (GRU),** overcome this problem by introducing a *long-term memory* and *forgetting*, or *memory resetting*.



Source: http://dprogrammer.org/rnn-lstm-gru

RNN Model in Python

```
# The LSTM architecture
model_lstm = Sequential()                                    50      1 (univariate)
model_lstm.add(LSTM(units=100, activation="tanh", input_shape=(n_steps, features)))
model_lstm.add(Dense(units=forecasting_horizon))
# Compiling the model                        7
model_lstm.compile(optimizer="RMSprop", loss="mse")

model_lstm.summary()
```




```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 lstm_1 (LSTM)                (None, 100)               40800

 dense_1 (Dense)              (None, 7)                 707

=================================================================
Total params: 41,507
Trainable params: 41,507
Non-trainable params: 0
_________________________________________________________________
```

4*(Features + LSTM_Unit + 1) * LSTM_Unit
= 4*(1 + 100 + 1) * 100

(LSTM_Unit + 1) * forecasting_horizon
= (100 + 1) * 7

RNN Model in Python

```
model_lstm.fit(X_train, y_train, epochs=100, batch_size=32)
```

```
Epoch 1/100
38/38 [==============================] - 7s 7ms/step - loss: 0.0196
Epoch 2/100
38/38 [==============================] - 0s 6ms/step - loss: 0.0040
Epoch 3/100
38/38 [==============================] - 0s 6ms/step - loss: 0.0034
Epoch 4/100
38/38 [==============================] - 0s 5ms/step - loss: 0.0026
Epoch 5/100
38/38 [==============================] - 0s 5ms/step - loss: 0.0024
```

X_test shape: (139, 50, 1)
y_test shape: (139, 7, 1)

```
predicted_stock_price = model_lstm.predict(X_test)
```

```
predicted_stock_price shape:  (139, 7)
```

```
for i in range(forecasting_horizon):
  print("Forecasting Horizon: {} ".format(i))
  return_mae(y_test[:,i],predicted_stock_price[:,i])
  print("")
```

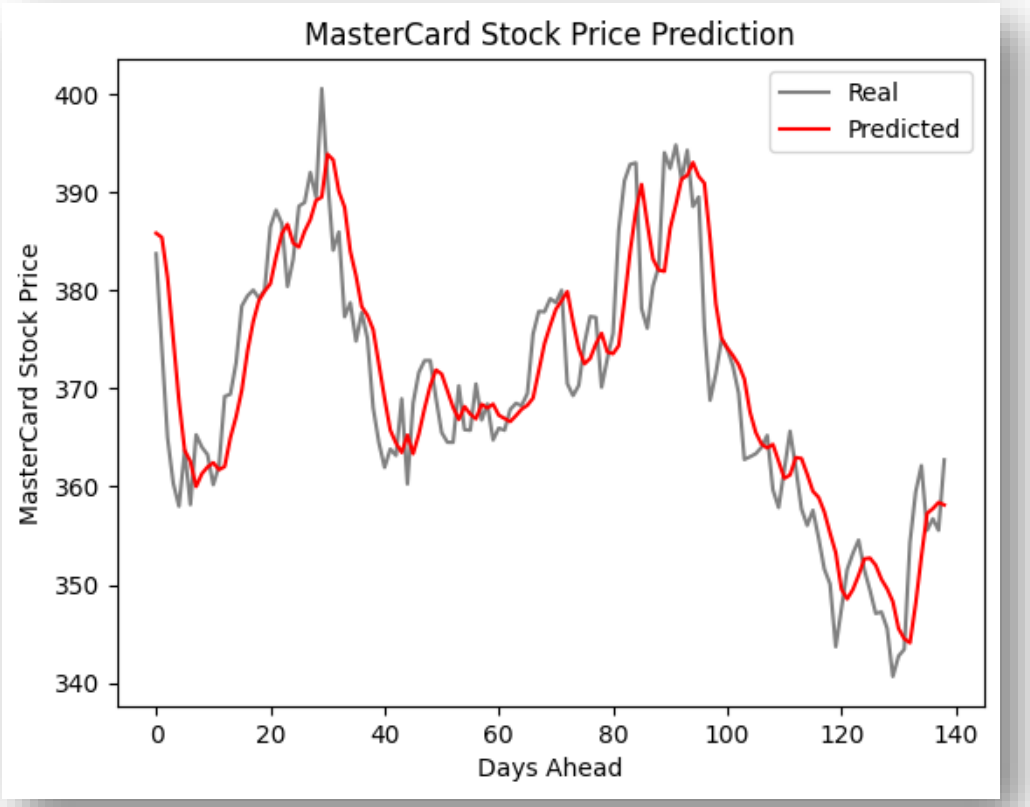
```
Forecasting Horizon: 0
Mean Absolute Error 4.42.

Forecasting Horizon: 1
Mean Absolute Error 5.95.

Forecasting Horizon: 2
Mean Absolute Error 8.16.

Forecasting Horizon: 3
Mean Absolute Error 7.51.

Forecasting Horizon: 4
Mean Absolute Error 8.39.

Forecasting Horizon: 5
Mean Absolute Error 8.59.

Forecasting Horizon: 6
Mean Absolute Error 10.14.
```

Forecasting Performance decreases when predicting further in the future

RNN Model in Python

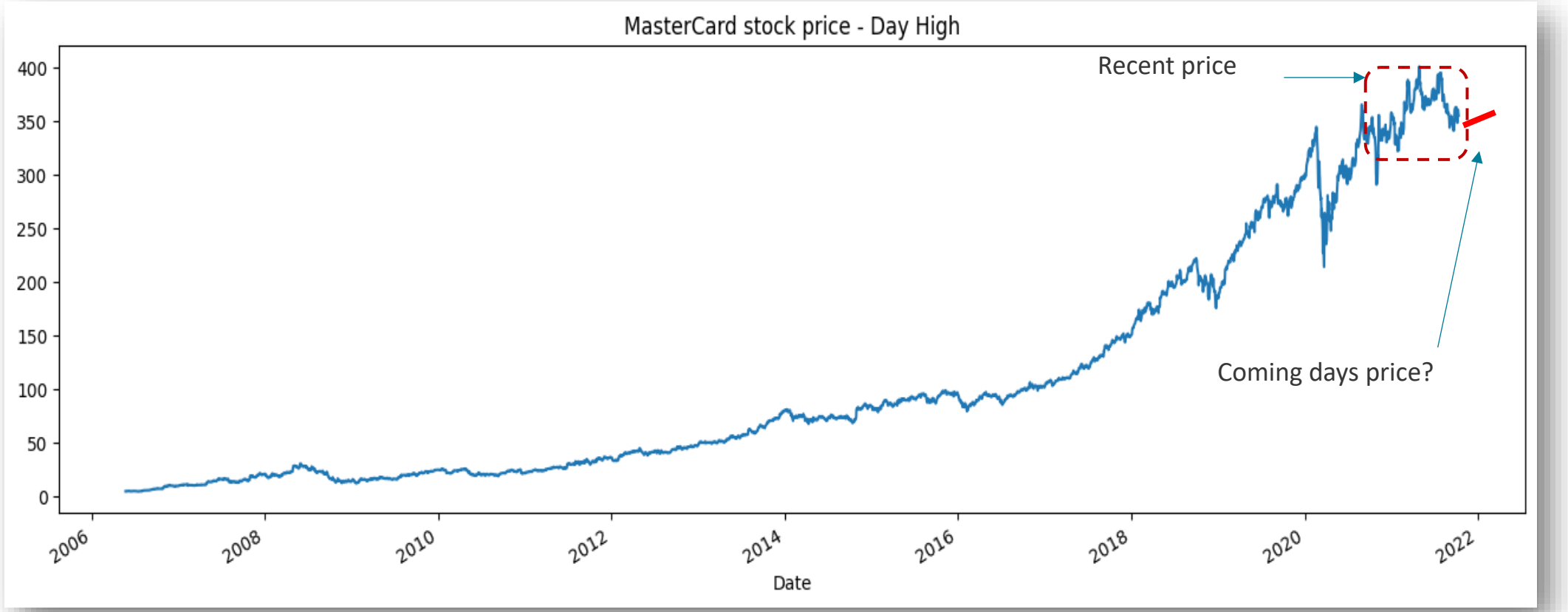
Visualize one testing sample and predicted values (7 days horizon)

Visualize test data and predicted values for one day forecasting horizon

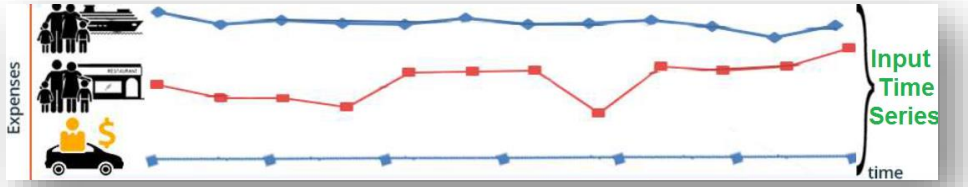


Model Deployment

Predict stock price based on recent price.



Multi-Variate Time Series Forecasting

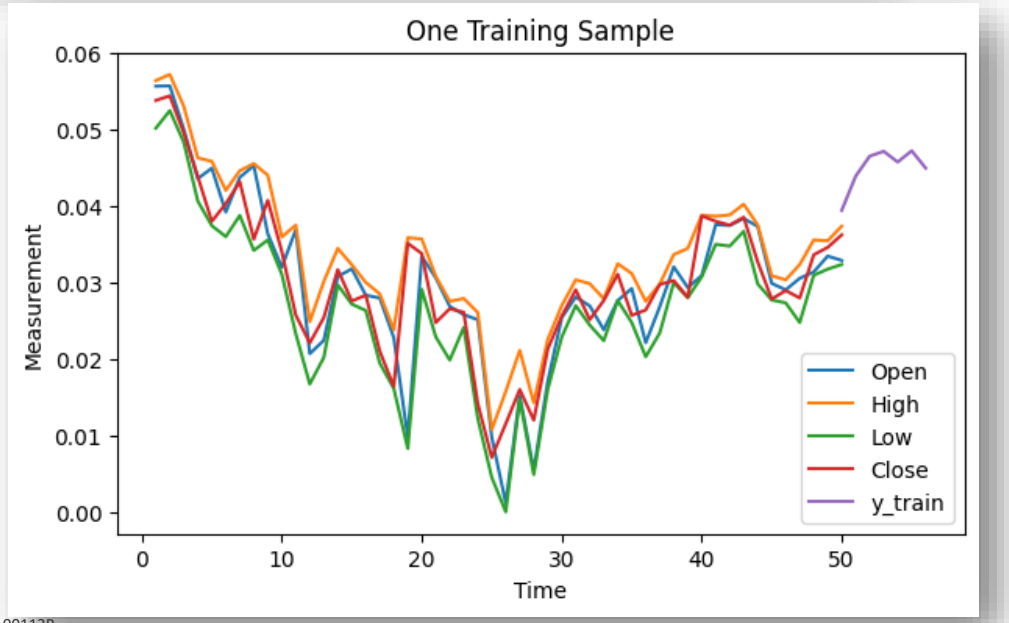


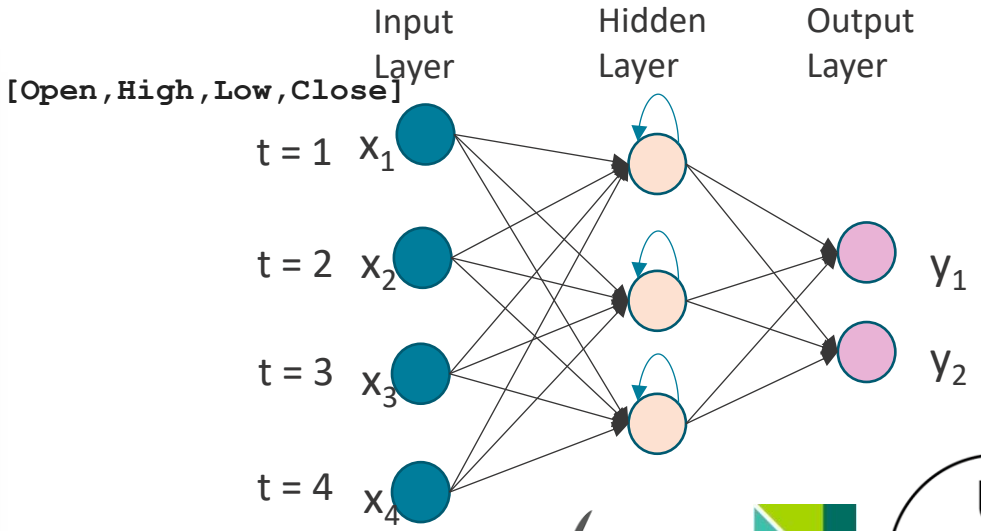
Date	Open	High	Low	Close	Volume
25/05/2006	3.7489667	4.2838687	3.7396638	4.2792172	395343000
26/05/2006	4.3071262	4.3480576	4.1033979	4.1796799	103044000
30/05/2006	4.1834002	4.1843304	3.9861838	4.0931644	49898000
31/05/2006	4.1257227	4.2196792	4.1257227	4.1806083	30002000
1/06/2006	4.1796782	4.4745719	4.1768872	4.4196863	62344000
2/06/2006	4.5117821	4.5303874	4.3527069	4.3713121	37253000
5/06/2006	4.3768952	4.5815537	4.3722438	4.5722508	37188000

Multi-variate Forecasting:
Future is predicted by on historical data of multiple time series

Forecasting Problem:
Predict the **High** price of Mastercard Stock of the coming 7 days based on historical data of **Open**, **High**, **Low** and **Close**.

```
X_train shape:
(1203, 50, 4)

y_train shape:
(1203, 7, 1) 0
```





Multi-Variate Time Series Forecasting (cont.)

```
# The LSTM architecture
model_lstm = Sequential()                                          50     4 (multivariate)
model_lstm.add(LSTM(units=100, activation="tanh", input_shape=(n_steps, features)))
model_lstm.add(Dense(units=forecasting_horizon))
# Compiling the model
model_lstm.compile(optimizer="RMSprop", loss="mse")

model_lstm.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 100)               42000

 dense (Dense)               (None, 7)                 707

=================================================================
Total params: 42,707
Trainable params: 42,707
Non-trainable params: 0
_________________________________________________________________
```

4*(Features + LSTM_Unit + 1) * LSTM_Unit
= 4*(4 + 100 + 1) * 100

Forecasting Performance

Based on One Time Series - **High**

Univariate

```
Forecasting Horizon: 0
Mean Absolute Error 4.42.

Forecasting Horizon: 1
Mean Absolute Error 5.95.

Forecasting Horizon: 2
Mean Absolute Error 8.16.

Forecasting Horizon: 3
Mean Absolute Error 7.51.

Forecasting Horizon: 4
Mean Absolute Error 8.39.

Forecasting Horizon: 5
Mean Absolute Error 8.59.

Forecasting Horizon: 6
Mean Absolute Error 10.14.
```

Multivariate

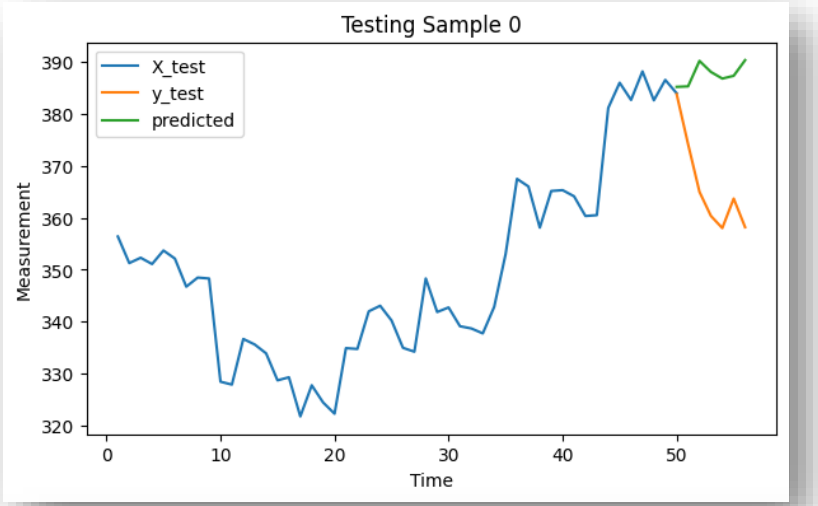
```
Forecasting Horizon: 0
Mean Absolute Error 3.98.

Forecasting Horizon: 1
Mean Absolute Error 6.62.

Forecasting Horizon: 2
Mean Absolute Error 8.11.

Forecasting Horizon: 3
Mean Absolute Error 7.77.

Forecasting Horizon: 4
Mean Absolute Error 7.68.

Forecasting Horizon: 5
Mean Absolute Error 8.04.

Forecasting Horizon: 6
Mean Absolute Error 9.87.
```

Based on Four Time Series – **Open**, **High**, **Low**, **Close**

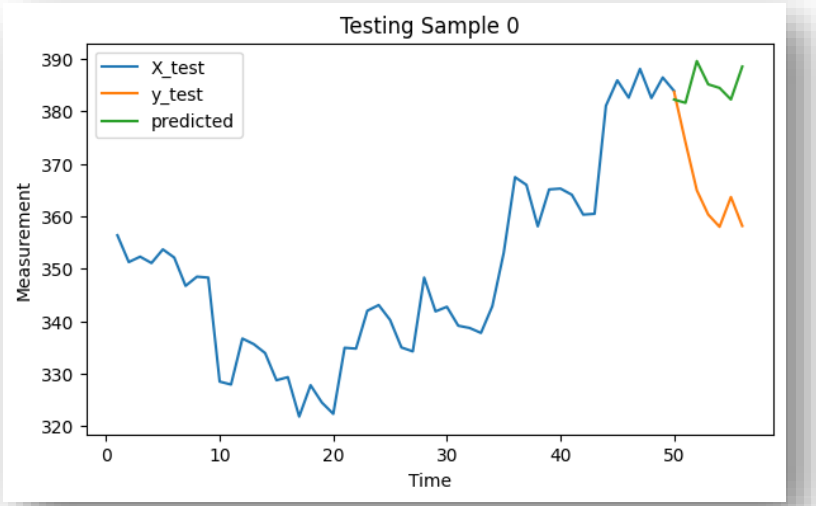
Which model is better?

Forecasting Performance

Univariate



Multivariate

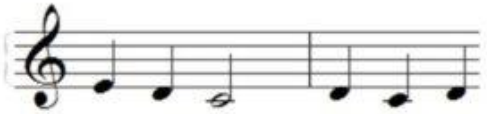


Which model is better?

What if we change the length of X_test (e.g. n_steps = 40 or 60)?

Visualize one testing sample and predicted values (7 days horizon)

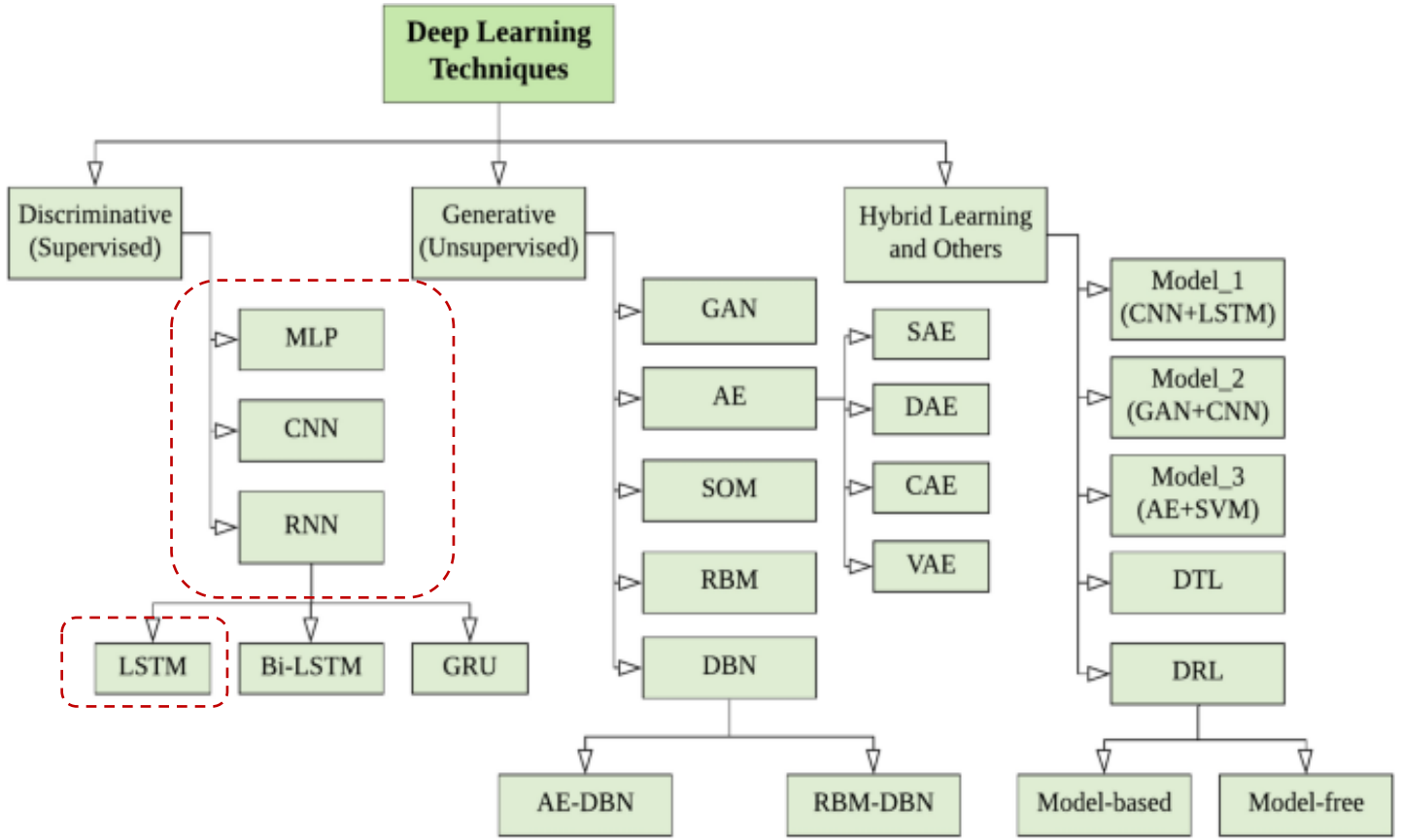
Other Applications of RNN

	x - input		y - output
Speech recognition		⟹	"Fuzzy Wuzzy was a bear. Fuzzy Wuzzy had no hair."
Music generation	Ø	⟹	
Sentiment classification	"Decent effort. The plot could have been better."	⟹	
DNA sequence analysis	ACTGTACCCATGTGACTGCCC	⟹	ACTGTACCCATGTGACTGCCC
Machine translation	"El que no arriesga, no gana."	⟹	"If you don't take risks, you cannot win."
Video activity recognition		⟹	Running
Name entity recognition	"Ygritte says Jon Snow knows nothing."	⟹	"Ygritte says Jon Snow knows nothing."

Source:https://datahacker.rs/003-rnn-architectural-types-of-different-recurrent-neural-networks/

Reflections on Deep Learning

- We have explored a new way of data analytics
- All **deep learning models** rely on discovering layers of data representation
- **Deep learning** and the "**new AI**" is one of the fastest growing fields of research
- New models and new techniques are being developed as we speak
- You need to keep on learning to stay on top of these developments
- We have only scratched the surface of a very deep stack of learning
- We hope you have enjoyed this experience!



In this lecture, we have covered:

- The concepts and architectures of RNN.
- Applications of RNN - LSTM in Time Series Forecasting

Summary