

<https://www.meetup.com/DSDTmtl>



Sept
29
2021

Data Science | Design | Technology

Data Science | Design | Technology

<https://www.meetup.com/DSDTmtl>

Sept
29
(2021)

Please, don't forget to
mute yourself





JL Maréchaux
DSDT Co-Organizer
(Google)



Christophe Pere
Chief Data Scientist
La Capitale Insurance

Agenda

DSDT Meetup - Sept 29, 2021

3:45 - 4:00 Arrival & Networking

4:00 - 4:15 News & Intro

4:15 - 5:15 Time series in financial domain: A deep learning approach

5:15 - 5:30: Virtual Snack & Networking

DSDT meetups in 2021



Monthly cadence, on Wednesdays.

Incredible sessions already planned. Contact us with your expectations & ideas.



ML
Validation

Reinforcement
Learning



Explainable
AI

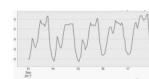
May 26

Rodent brain
& NN

Aug 25



RNN & Time
Series



Your meetup,
your ideas.



<http://bit.ly/DSDTsurvey2021>

Cooperathon 2021



Sept 28 - Dec 1: The largest Open Innovation challenge in Canada (cooperathon.ca)

Community & Training Partners



McGill | Engine



FASKEN



<MTL> Connect, October 12-17



Virtual Meetups

Until we can do in-person events again in Montreal...

Past (and future) presentations available on Slideshare.

[slideshare.net/DSDT_MTL](https://www.slideshare.net/DSDT_MTL)



DSDT Meetups: In-person events

Back to normal soon at [Notman House](#)



Christophe Pere

Chief Data Scientist
La Capitale Insurance



Time series in financial domain

A deep learning approach

Content

- Who am I
- Time Series
- RNN Family
- Applications with real data
- Discussion
- Conclusion
- Future

Who am I

- French guy
- PhD in Astrophysics
- Researcher since 2016
 - Automotive market
 - Autonomous vehicle
 - NLP
 - Synthetic data (imbalanced data)
 - Knowledge Representation
 - Reasoning
- Passion for AI
- Passion for Quantum

Time Series

*In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean tides, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.*

Wikipedia¹

¹ https://en.wikipedia.org/wiki/Time_series

Time Series

*In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean tides, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.*

Wikipedia¹

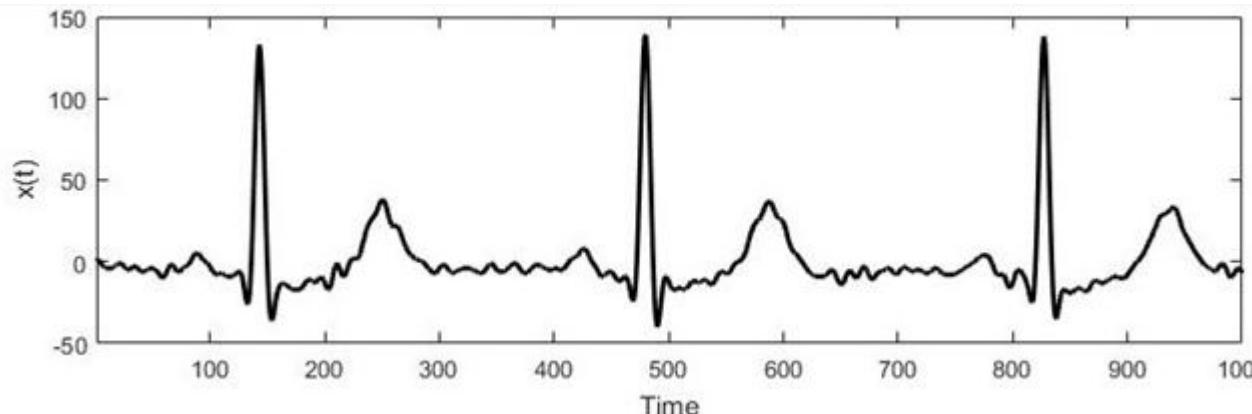
Ok, but in practice?

¹ https://en.wikipedia.org/wiki/Time_series

Time Series

In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean [tides](#), counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.

Wikipedia¹



Electrocardiogram²

¹https://en.wikipedia.org/wiki/Time_series

²Čepulionis Paulius, Lukoševičiūtė Kristina Electrocardiogram time series forecasting and optimization using ant colony optimization algorithm. Mathematical Models in Engineering, Vol. 2, Issue 1, 2016, p. 69-77

Time Series

In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean **tides**, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.

Wikipedia¹



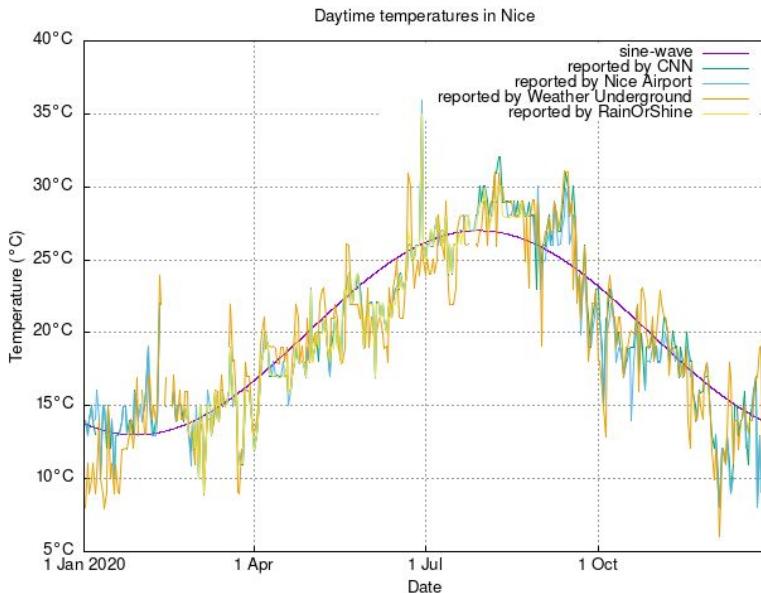
¹ https://en.wikipedia.org/wiki/Time_series

² https://ionides.github.io/531w16/final_project/Project13/final_project.html

Time Series

In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean [tides](#), counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.

Wikipedia¹



Temperatures in Nice city, France²

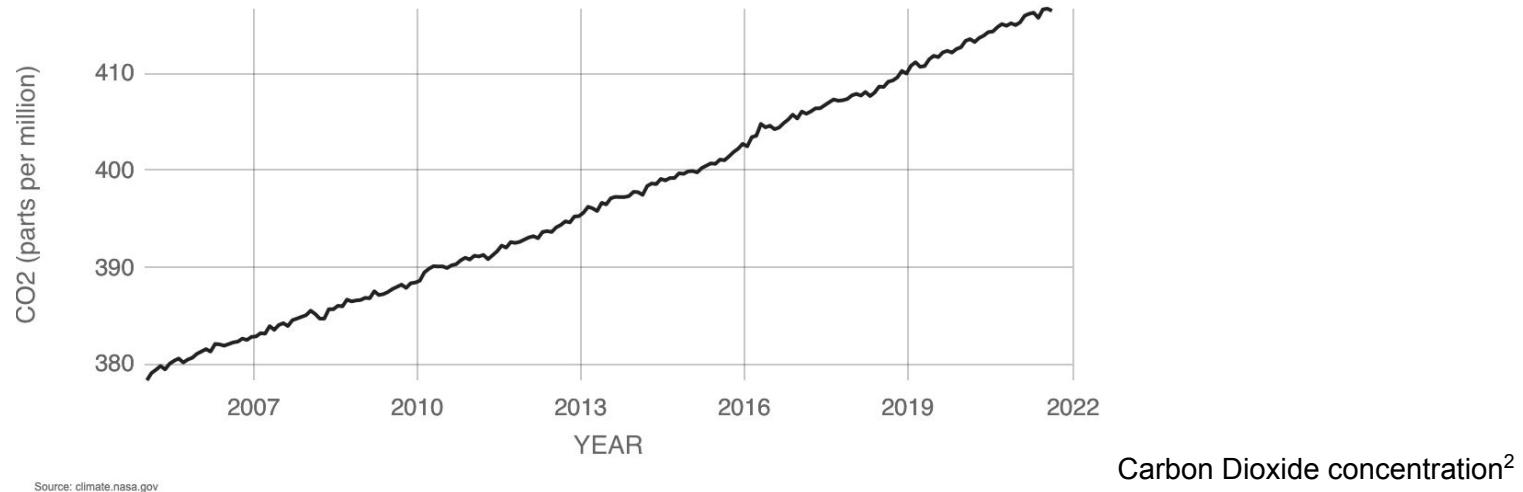
¹ https://en.wikipedia.org/wiki/Time_series

² <https://www.w3.org/People/Bos/Nice/tempgraph>

Time Series

In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean **tides**, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.

Wikipedia¹



¹ https://en.wikipedia.org/wiki/Time_series

² <https://climate.nasa.gov/vital-signs/carbon-dioxide/>

Time Series

*In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean tides, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.*

Wikipedia¹

Is that all? Could we use other kind of data with another representation of “time”?

¹ https://en.wikipedia.org/wiki/Time_series

Time Series

*In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean tides, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.*

Wikipedia¹

Is that all? Could we use other kind of data with another representation of “time”?

What do we call *Time* ?

¹ https://en.wikipedia.org/wiki/Time_series

Time Series

*In mathematics, a **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data. Examples of time series are heights of ocean tides, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.*

Wikipedia¹

Is that all? Could we use other kind of data with another representation of “time”?

What do we call *Time* ?

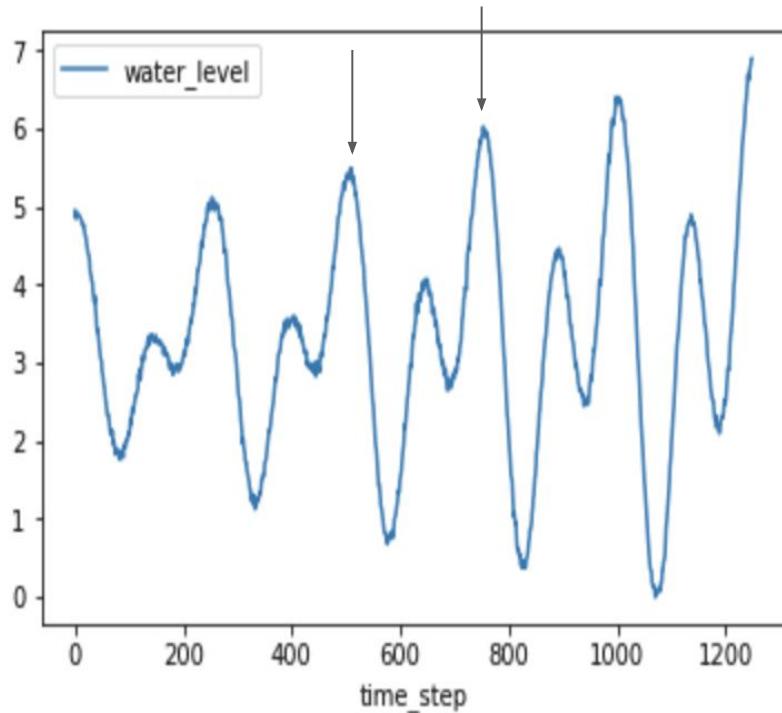
¹ https://en.wikipedia.org/wiki/Time_series

Time Series



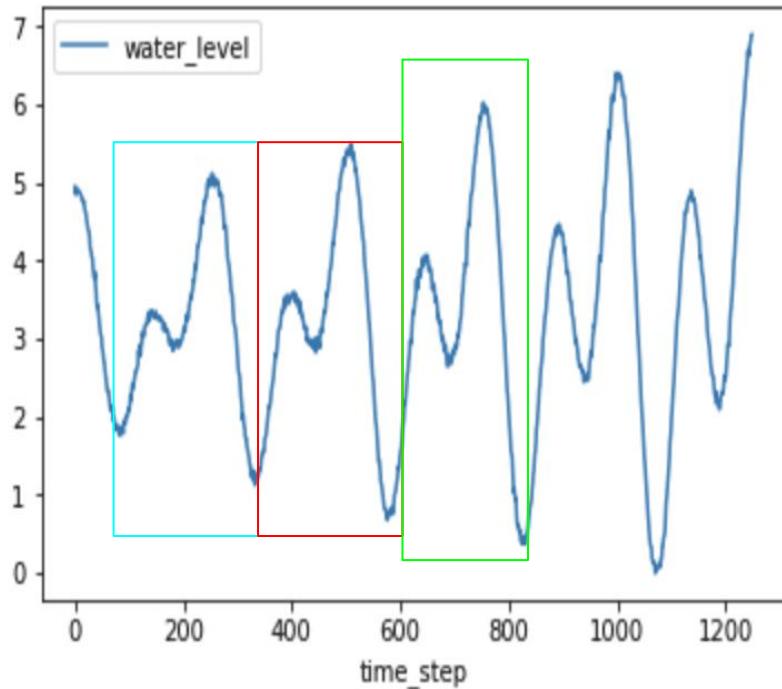
Time Series - Composition

Autocorrelation



Time Series - Composition

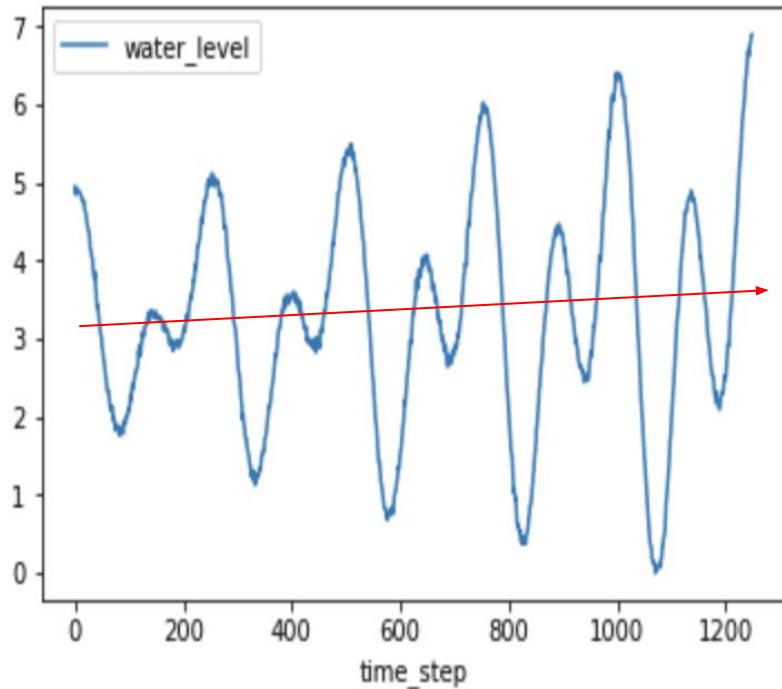
Seasonality



Time Series - Composition

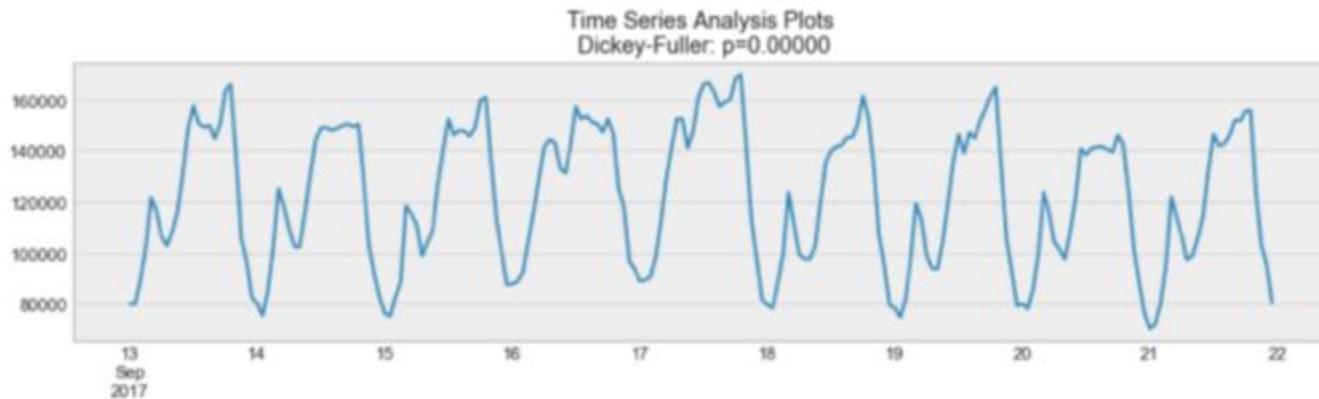
Trend

Movement of a series
(high and low values)
over a long period of
time



Time Series

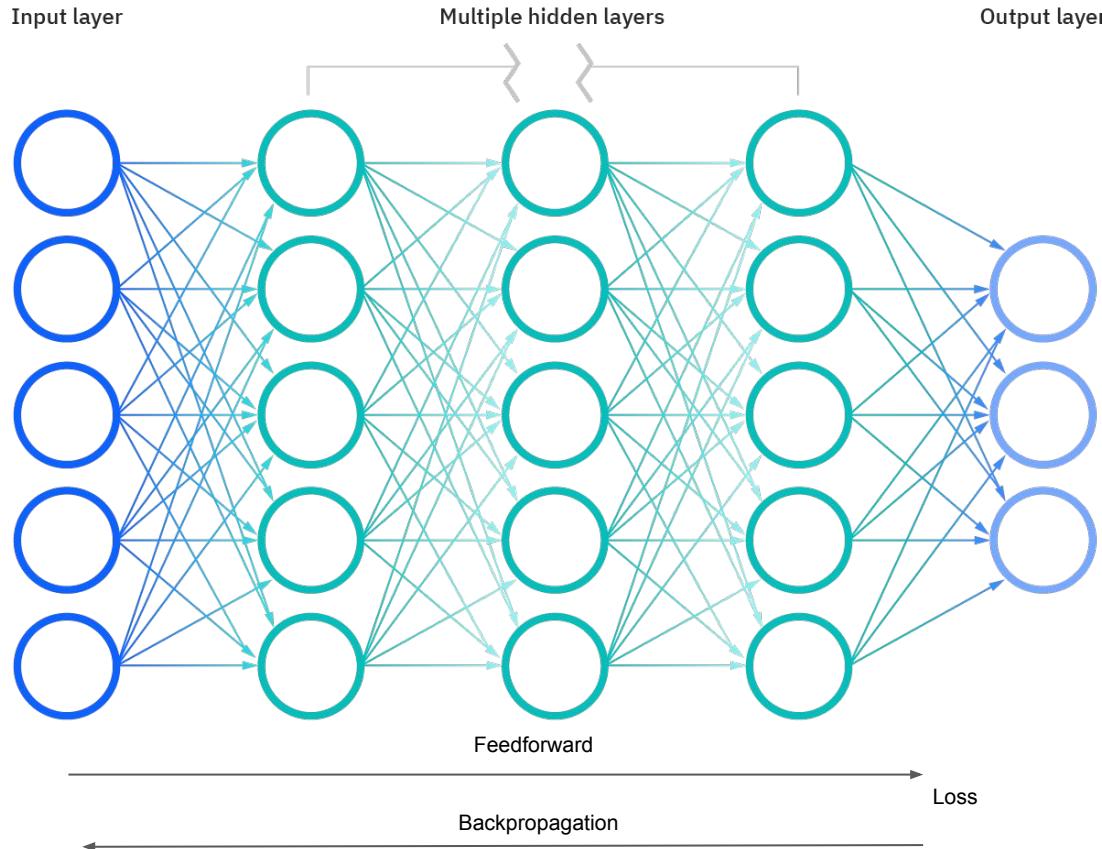
Stationarity



Constant mean
Constant variance
Covariance independent of time

RNN Family

Deep neural network



RNN Family

Sequence model

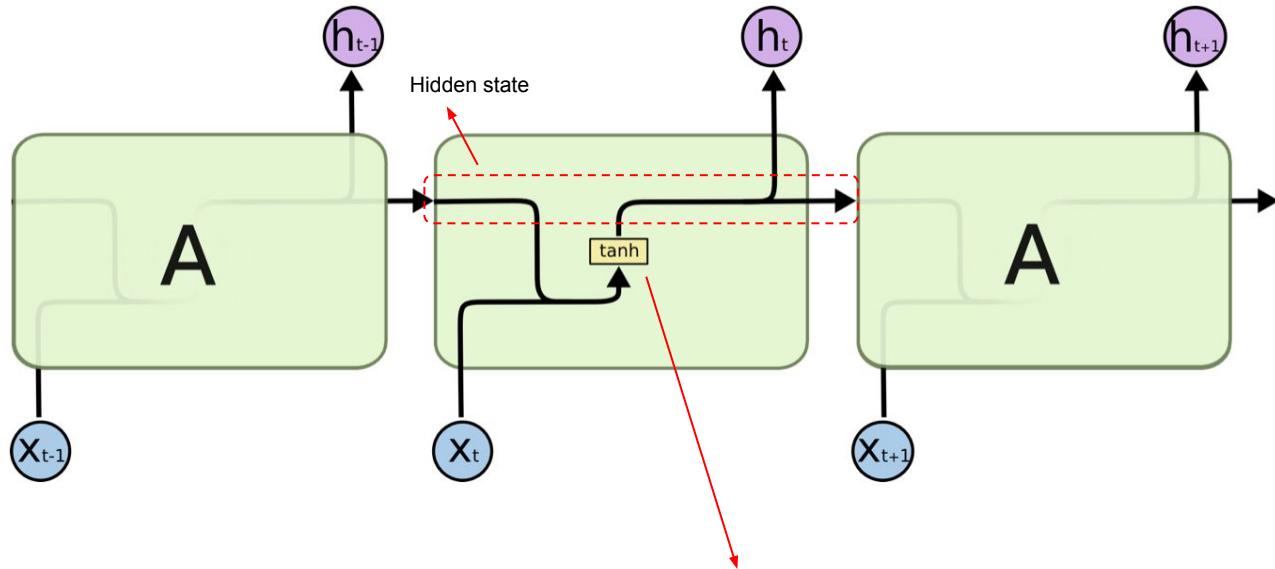
- text
- video
- sound

Persist information

- Memory
- Hidden state

RNN Family

Simple RNN



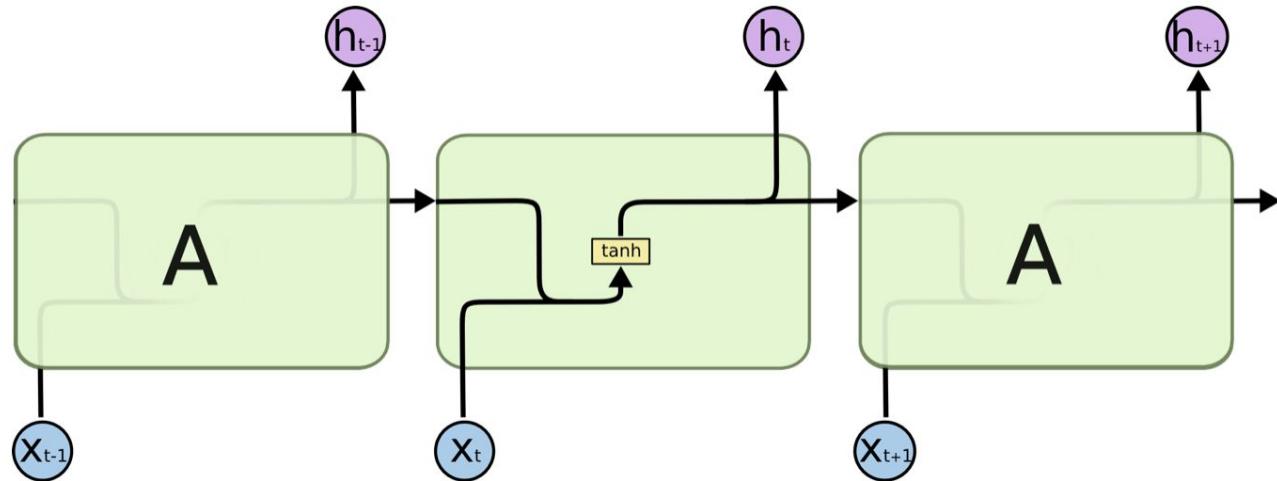
Advantage:

- Short-term memory

tanh has a range of -1, 1 allows to keep values coherent

RNN Family

Simple RNN

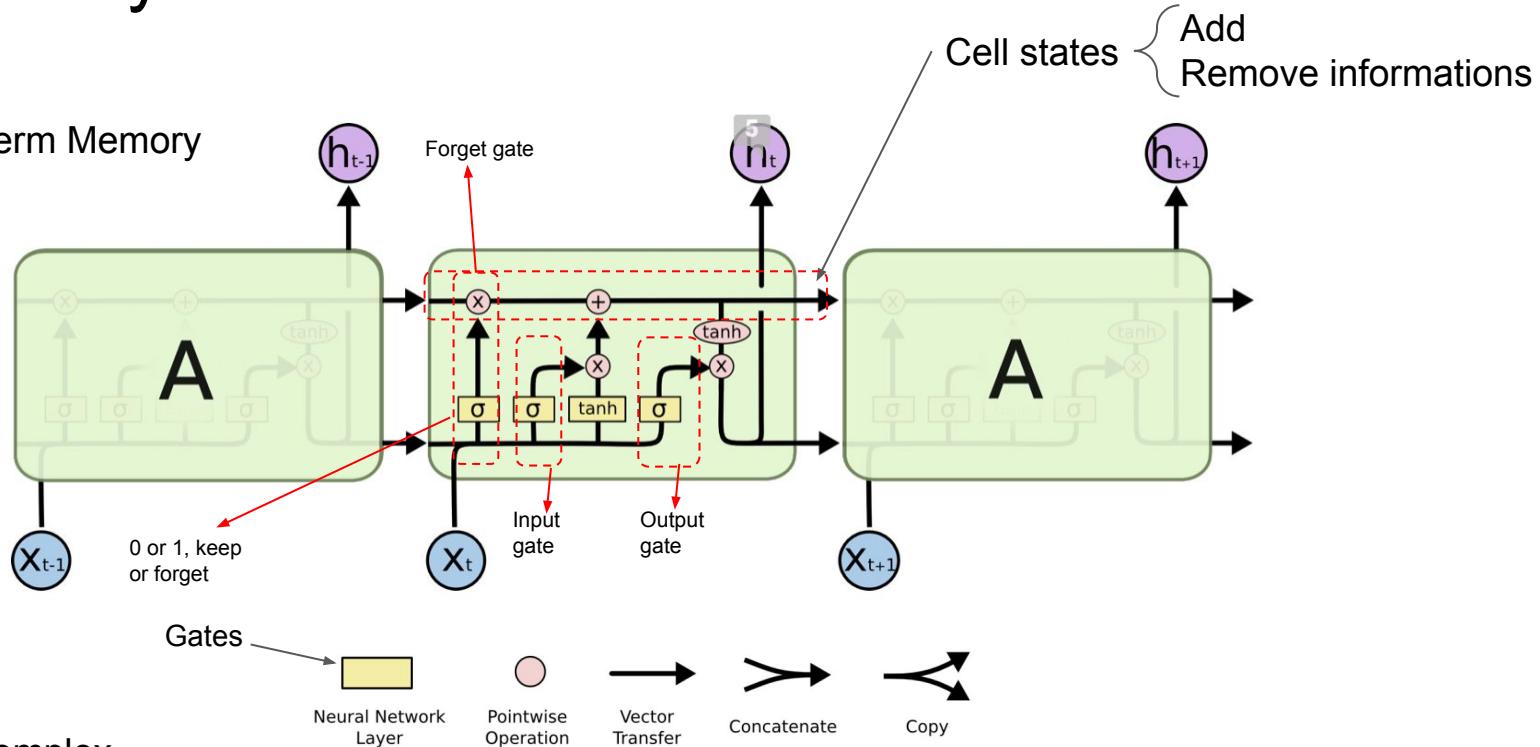


Problems:

- Vanishing gradient
- Exploding gradient
- Long memory (only remember the previous output)

RNN Family

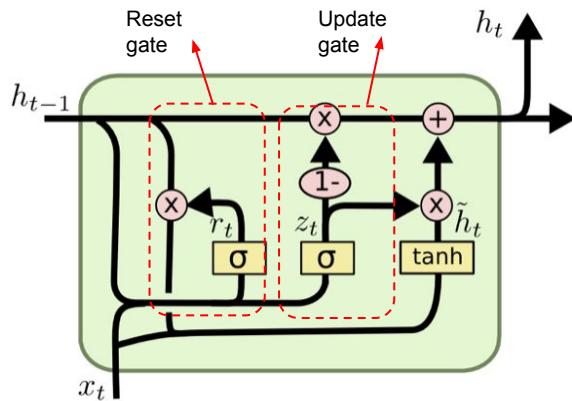
Long Short-Term Memory
(LSTM)



A little bit more complex...

RNN Family

Gated Recurrent Unit (GRU)



Combines the forget and input gates
into a single “update gate”

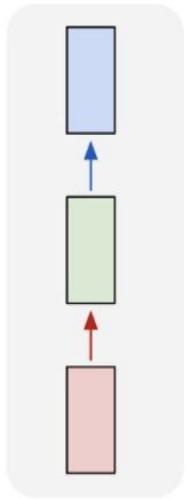
RNN Family

Applications:

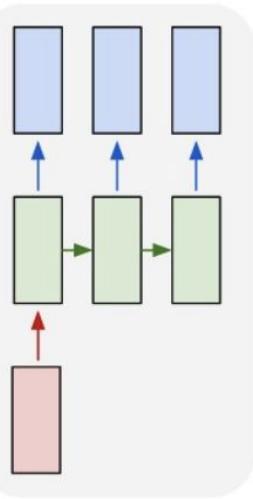
- Prediction problems (ex: forecasting)
- Language modelling
- Text Generation
- Machine Translation
- Speech Recognition
- Image description generation
- Video Tagging
- Text summarization
- Call center Analysis
- Face detection
- OCR
- Image Recognition
- Music Generation
- ...

RNN Family - Architectures

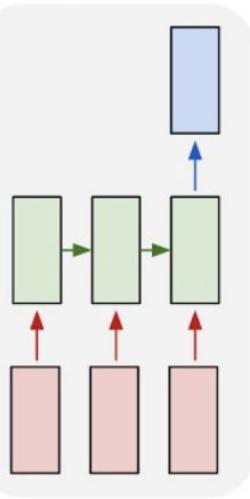
one to one



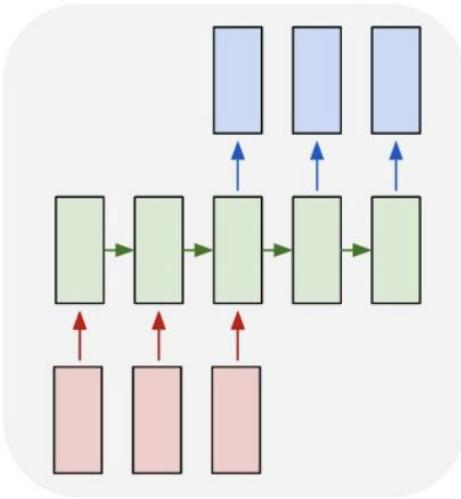
one to many



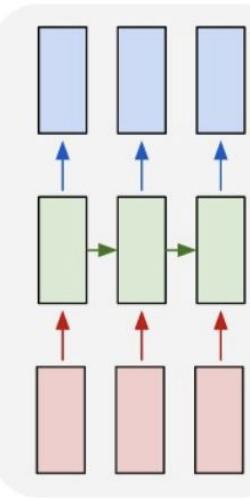
many to one



many to many



many to many



Classical
neural
network

Music
Generation

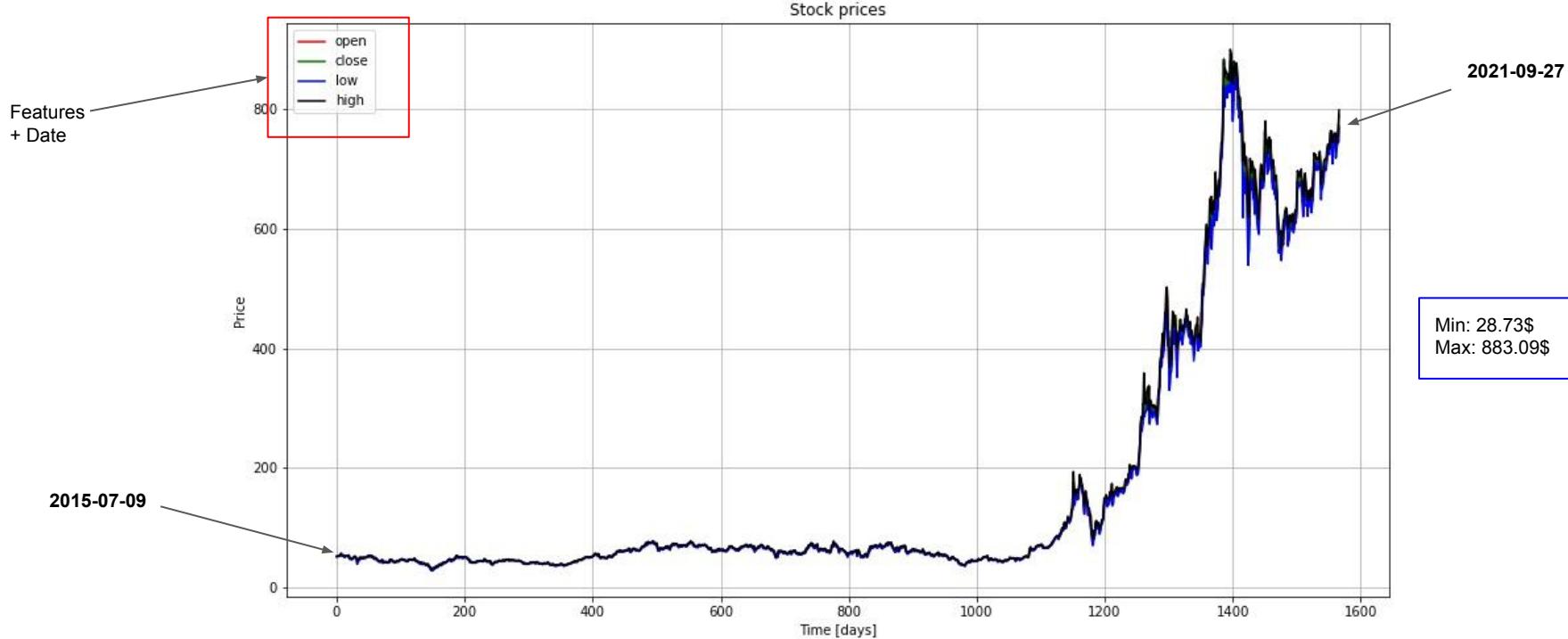
Sentiment
analysis

Machine
Translation

Name entity
recognition

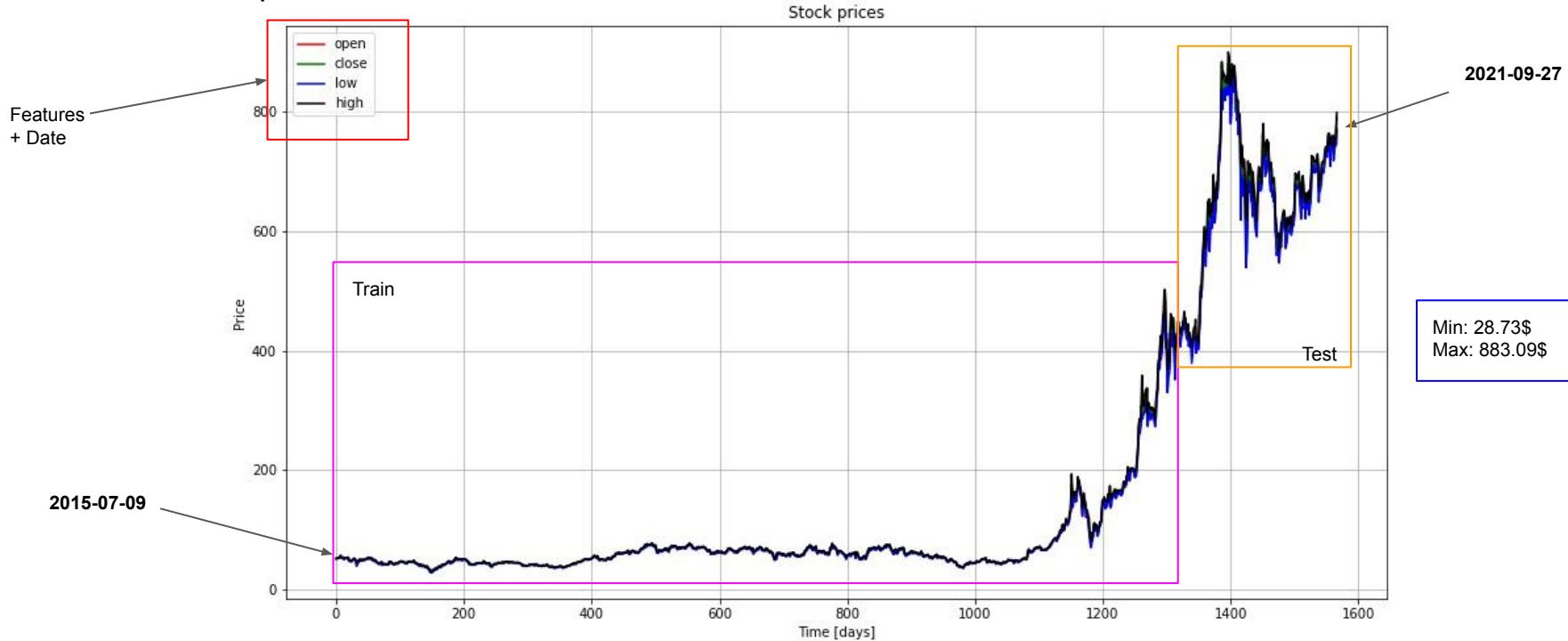
Application

TESLA stock prices



Application

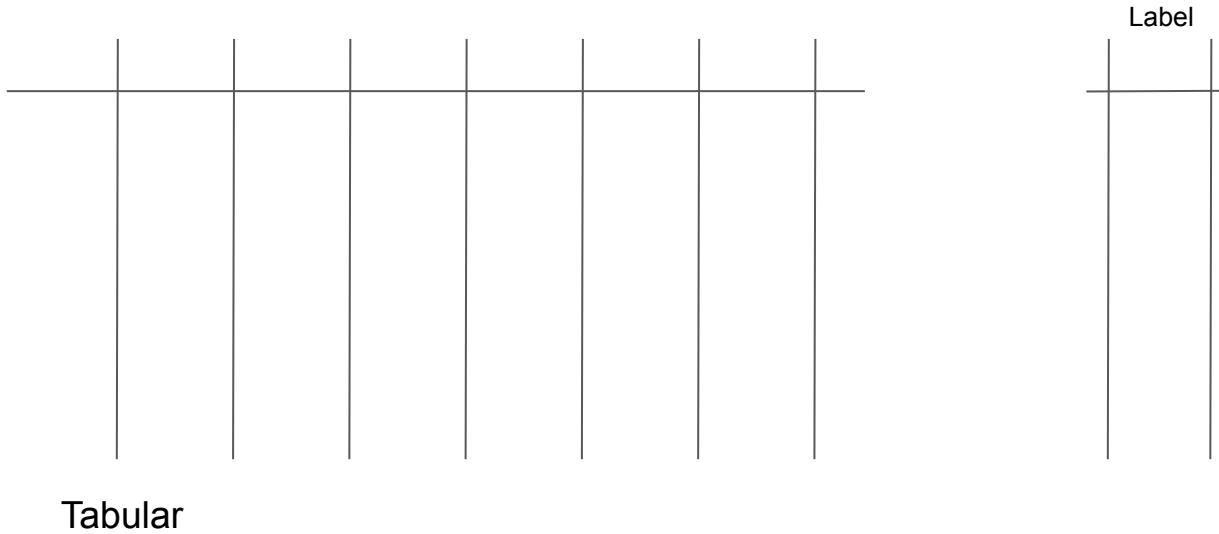
TESLA stock prices



Application

Forecasting?

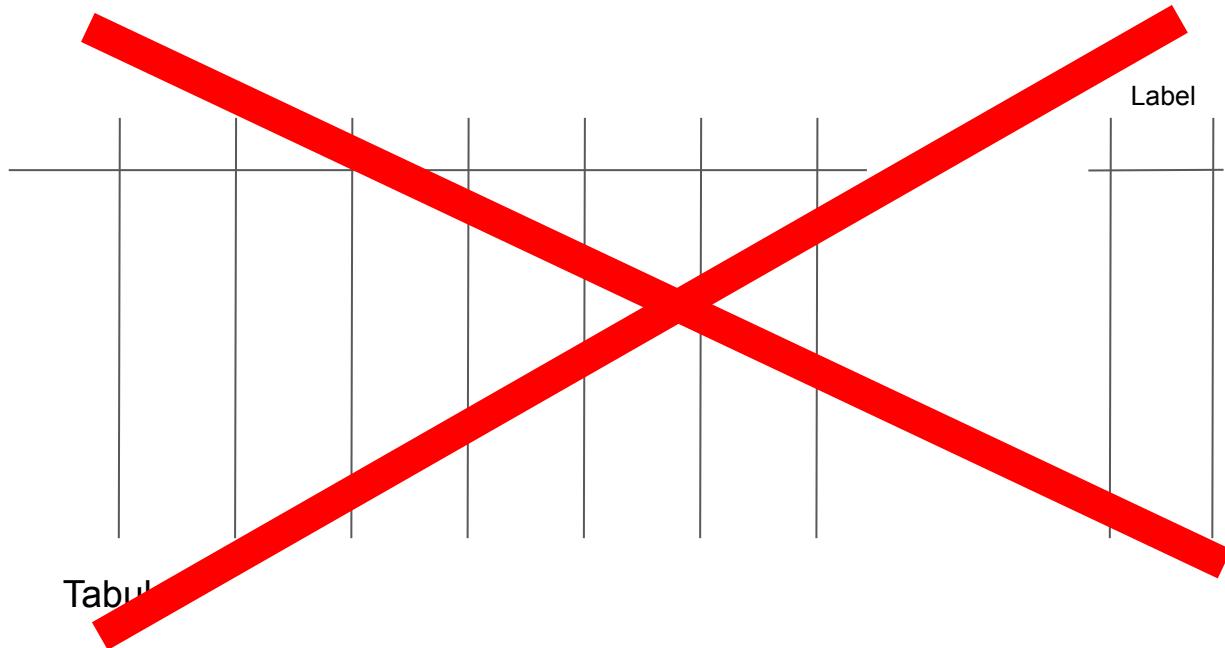
How to use the
data?



Application

Forecasting?

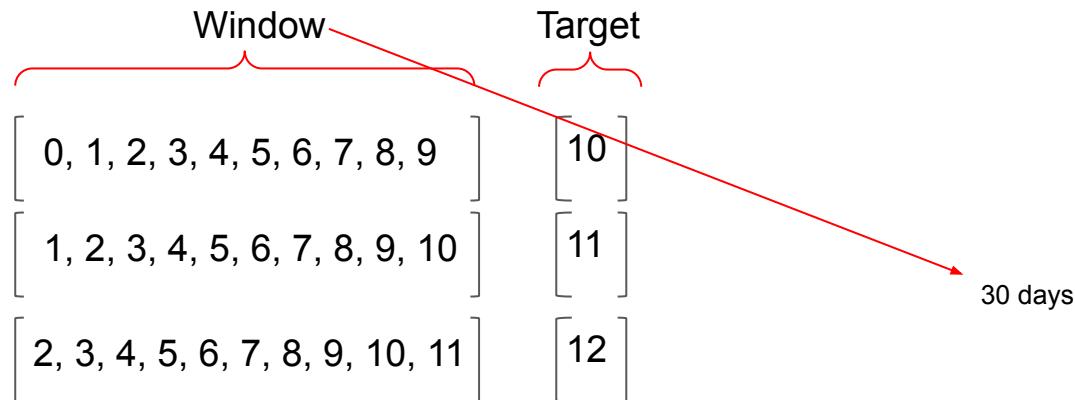
How to use the
data?



Application

Forecasting?

How to use the data?



Process to do for:

- Train
- Test

Normalization of data points:

- Subtract the mean computed on the training set and divide by the standard deviation
- Or
- Scale the data between 0 and 1

Application

Forecasting?

A little bit of code

RNN model

Visualization of
the loss for each
epoch

```
model_name = "rnn.keras"

inputs = keras.Input(shape=(X_train.shape[1],1))
x = layers.SimpleRNN(250, recurrent_dropout=0.2)(inputs)
outputs = layers.Dense(1)(x)
model = keras.Model(inputs, outputs)

callbacks = [
    keras.callbacks.ModelCheckpoint(model_name,
                                    save_best_only=True)
]
model.compile(optimizer="NAdam", loss="mse", metrics=["mae"])
history = model.fit(X_train, y_train,
                     epochs=50,
                     validation_split=0.2,
                     callbacks=callbacks, verbose = 0)

predicted_stock_price = eval_model(model_name, X_test, y_test)

loss = history.history["mae"]
val_loss = history.history["val_mae"]
epochs = range(1, len(loss) + 1)
plt.figure(figsize=(12,8))
plt.plot(epochs, loss, "bo", label="Training MAE")
plt.plot(epochs, val_loss, "b", label="Validation MAE")
plt.title("Training and validation MAE SimpleRNN + dropout")
plt.grid(True)
plt.legend()
plt.show()
```

Only this part will change

One neuron

Save the best model

Will compute the MAE

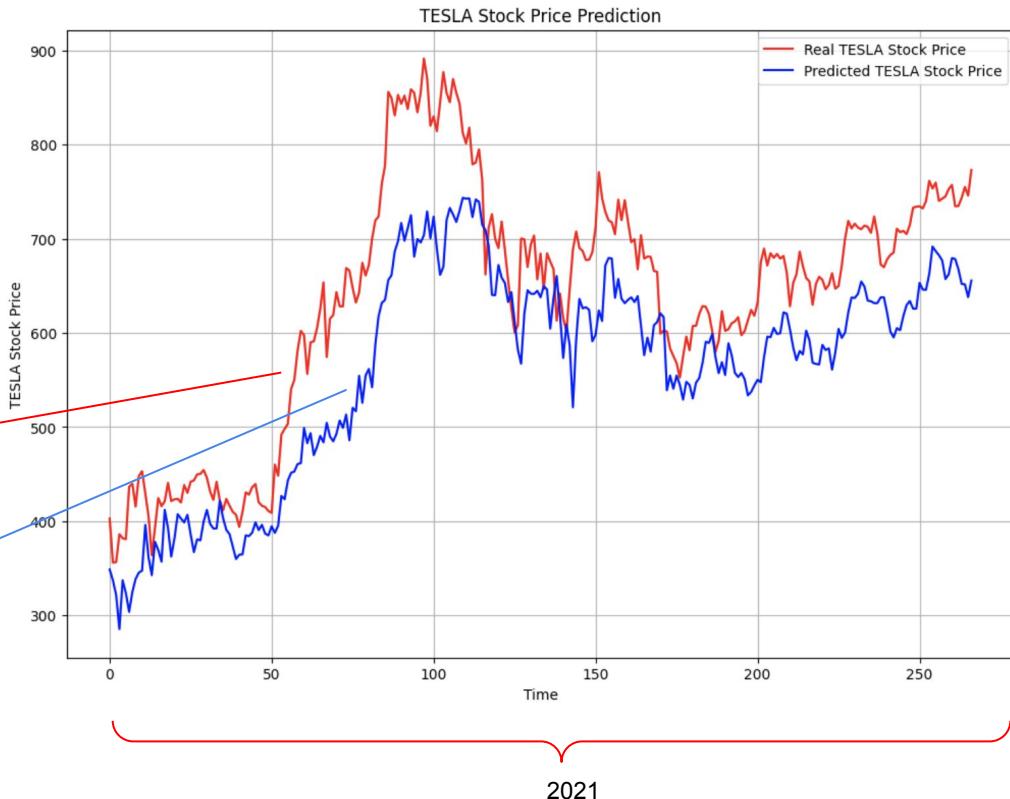
Application - Results

- First we need a **baseline**
- Simple neural network (50 neurons)
- Test MAE: 75.24

Computed on the denormalized data

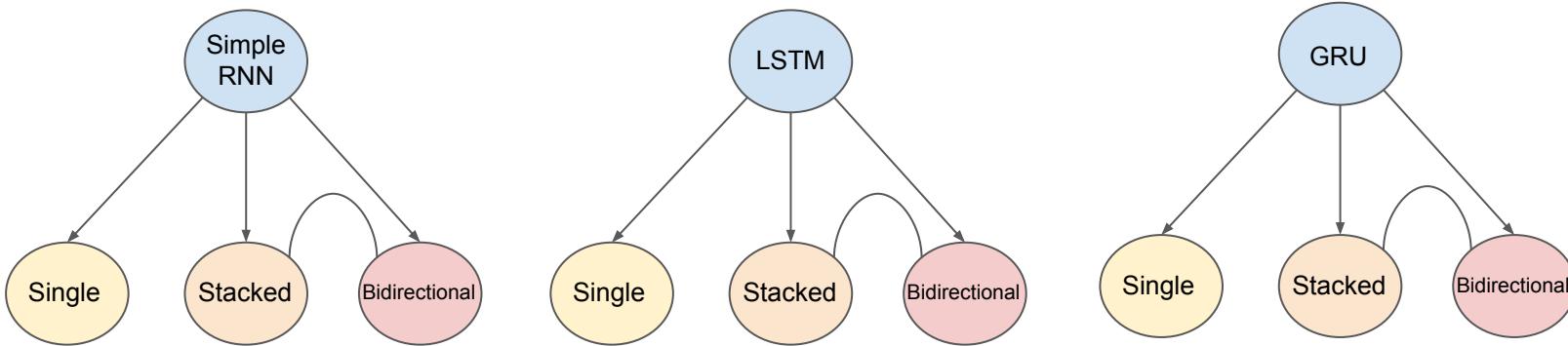
Real close value

Predict close value



Application - Results

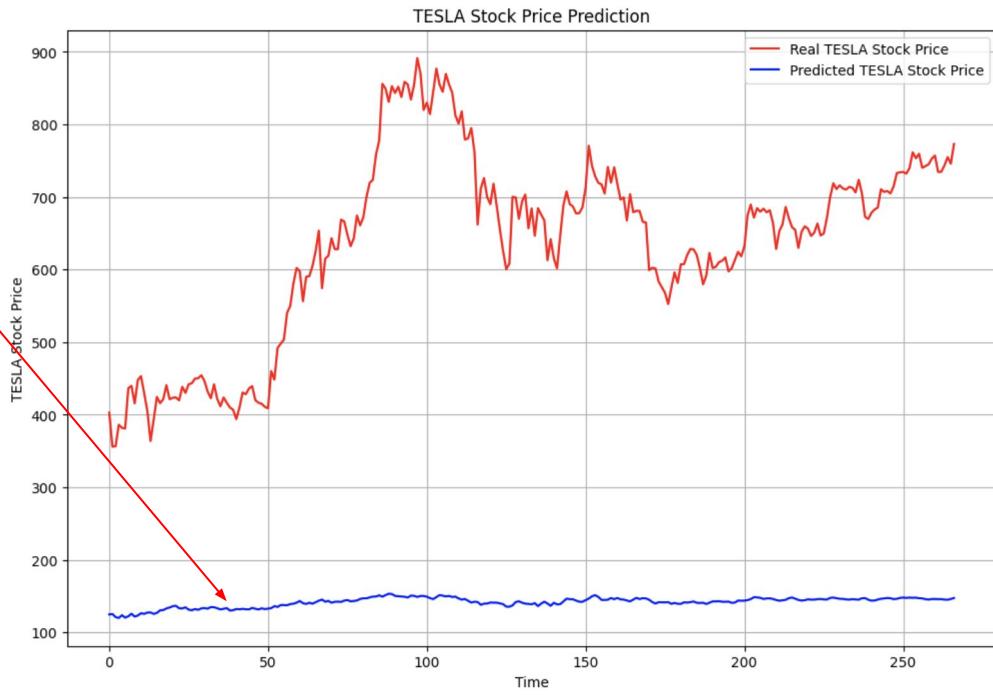
Models tested



Variations with
recurrent dropout
have been tested

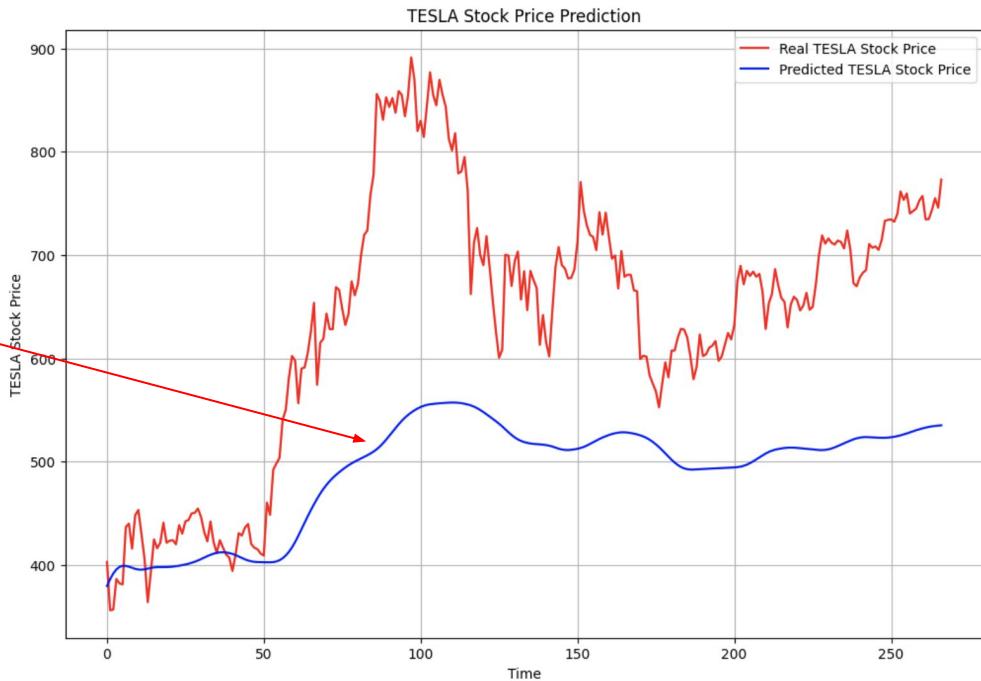
Application - Results

Models	MAE
SimpleRNN	76.46
Stacked SimpleRNN	448.94
Bidirectional SimpleRNN	235.19
Stacked Bidirectional SimpleRNN	494.51
LSTM	22.27
Stacked LSTM	146.99
Bidirectional LSTM	19.20
Stacked Bidirectional LSTM	124.32
GRU	20.70
Stacked GRU	103.21
Bidirectional GRU	18.94
Stacked Bidirectional GRU	66.88
Baseline (NN)	75.24



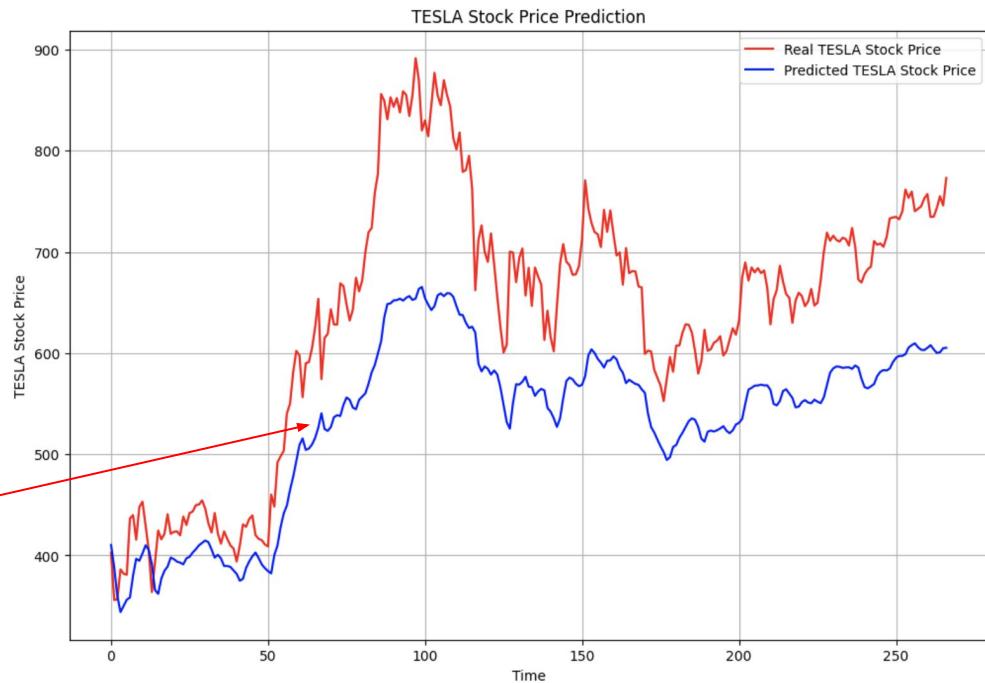
Application - Results

Models	MAE
SimpleRNN	76.46
Stacked SimpleRNN	448.94
Bidirectional SimpleRNN	235.19
Stacked Bidirectional SimpleRNN	494.51
LSTM	22.27
Stacked LSTM	146.99
Bidirectional LSTM	19.20
Stacked Bidirectional LSTM	124.32
GRU	20.70
Stacked GRU	103.21
Bidirectional GRU	18.94
Stacked Bidirectional GRU	66.88
Baseline (NN)	75.24



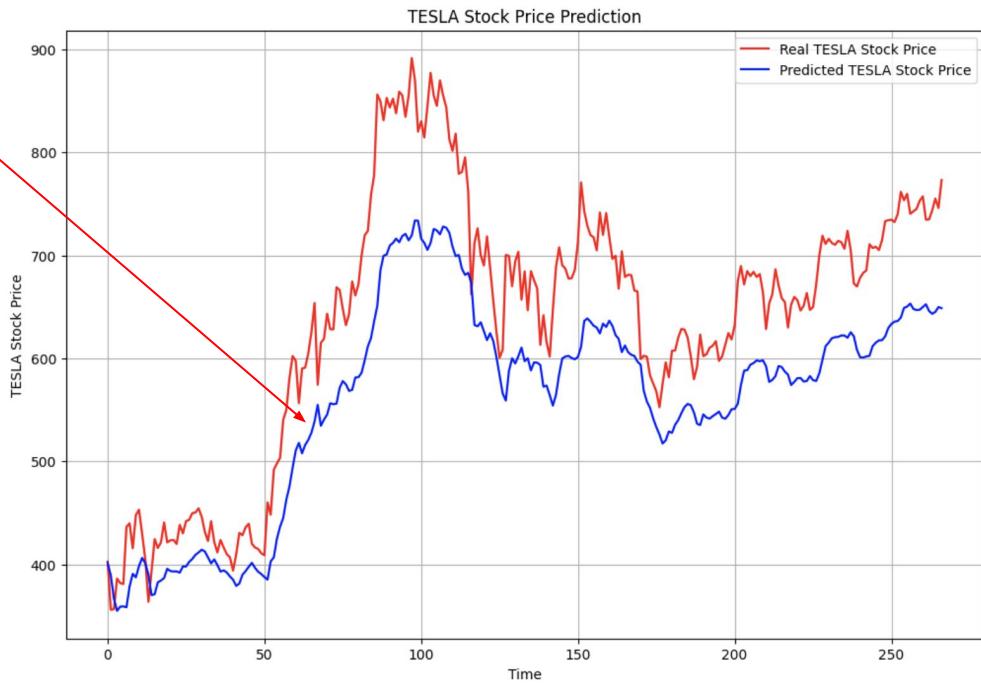
Application - Results

Models	MAE
SimpleRNN	76.46
Stacked SimpleRNN	448.94
Bidirectional SimpleRNN	235.19
Stacked Bidirectional SimpleRNN	494.51
LSTM	22.27
Stacked LSTM	146.99
Bidirectional LSTM	19.20
Stacked Bidirectional LSTM	124.32
GRU	20.70
Stacked GRU	103.21
Bidirectional GRU	18.94
Stacked Bidirectional GRU	66.88
Baseline (NN)	75.24



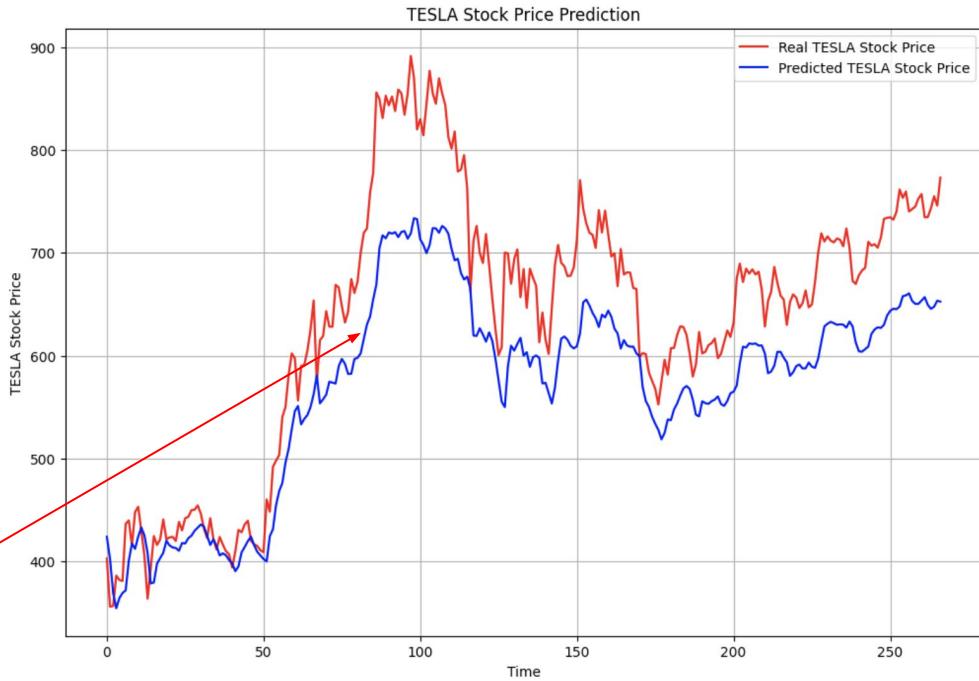
Application - Results

Models	MAE
SimpleRNN	76.46
Stacked SimpleRNN	448.94
Bidirectional SimpleRNN	235.19
Stacked Bidirectional SimpleRNN	494.51
LSTM	22.27
Stacked LSTM	146.99
Bidirectional LSTM	19.20
Stacked Bidirectional LSTM	124.32
GRU	20.70
Stacked GRU	103.21
Bidirectional GRU	18.94
Stacked Bidirectional GRU	66.88
Baseline (NN)	75.24



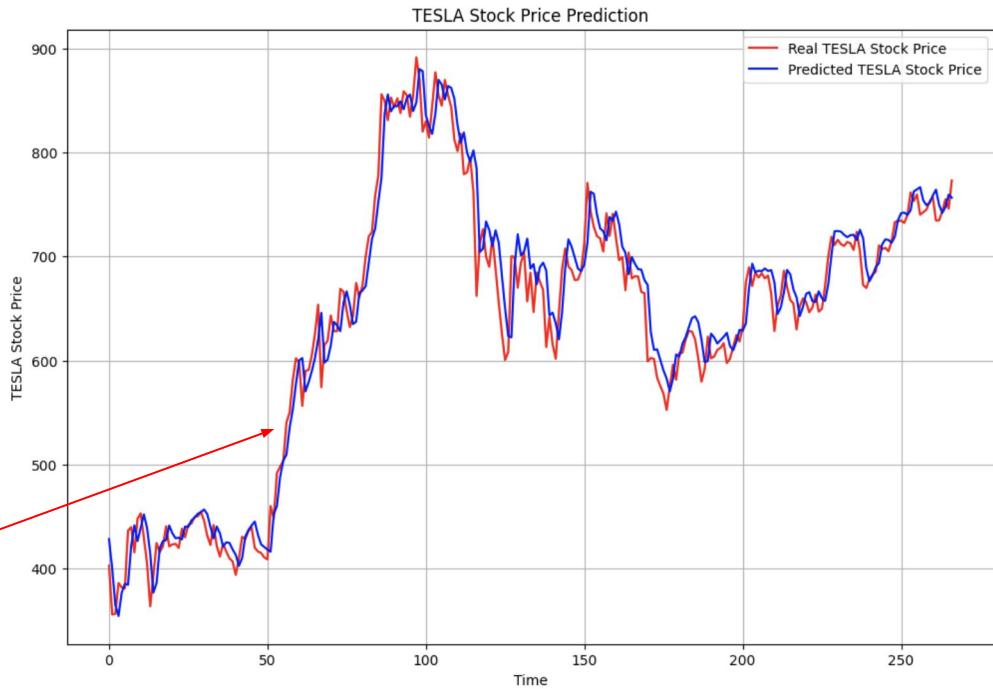
Application - Results

Models	MAE
SimpleRNN	76.46
Stacked SimpleRNN	448.94
Bidirectional SimpleRNN	235.19
Stacked Bidirectional SimpleRNN	494.51
LSTM	22.27
Stacked LSTM	146.99
Bidirectional LSTM	19.20
Stacked Bidirectional LSTM	124.32
GRU	20.70
Stacked GRU	103.21
Bidirectional GRU	18.94
Stacked Bidirectional GRU	66.88
Baseline (NN)	75.24



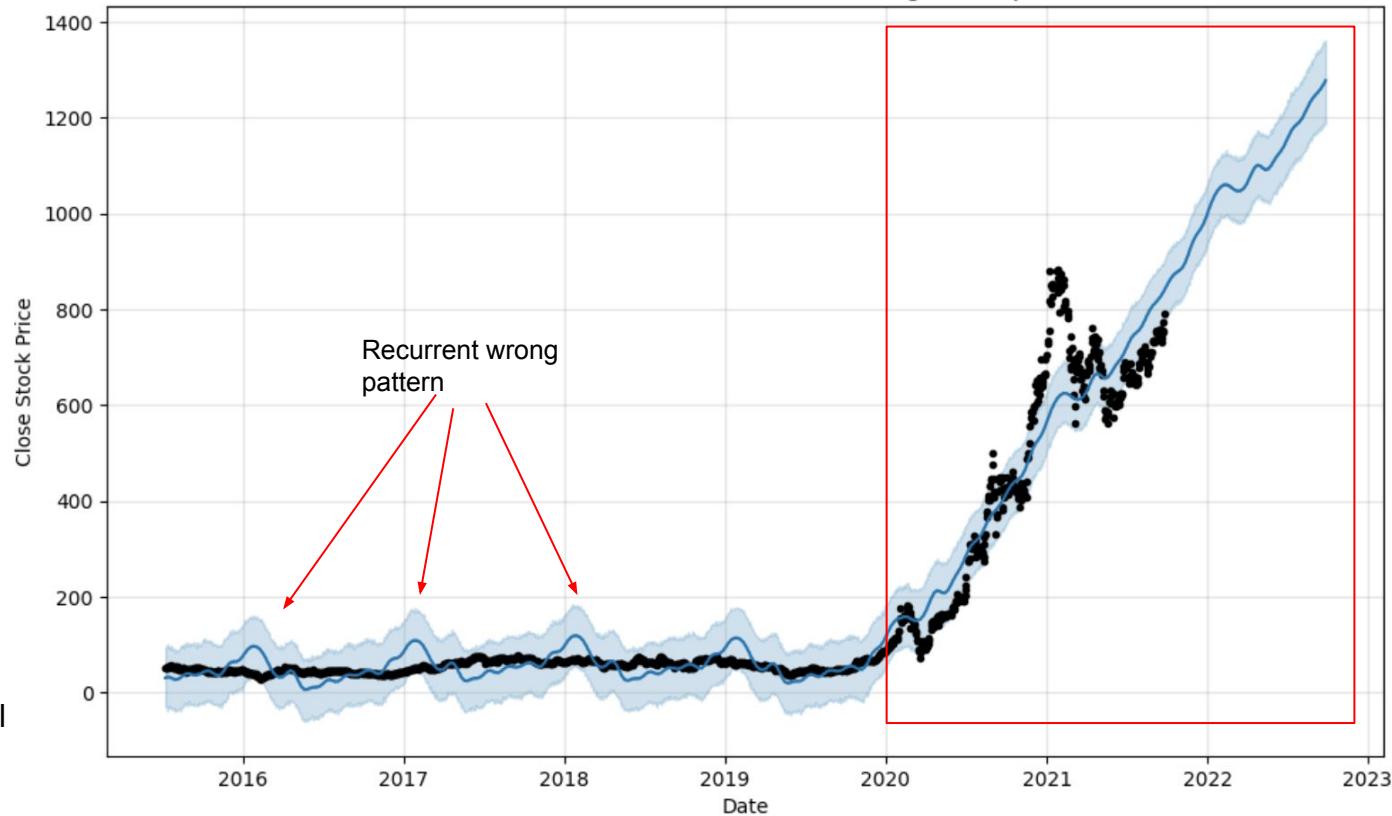
Application - Results

Models	MAE
SimpleRNN	76.46
Stacked SimpleRNN	448.94
Bidirectional SimpleRNN	235.19
Stacked Bidirectional SimpleRNN	494.51
LSTM	22.27
Stacked LSTM	146.99
Bidirectional LSTM	19.20
Stacked Bidirectional LSTM	124.32
GRU	20.70
Stacked GRU	103.21
Bidirectional GRU	18.94
Stacked Bidirectional GRU	66.88
Baseline (NN)	75.24



Application - Results

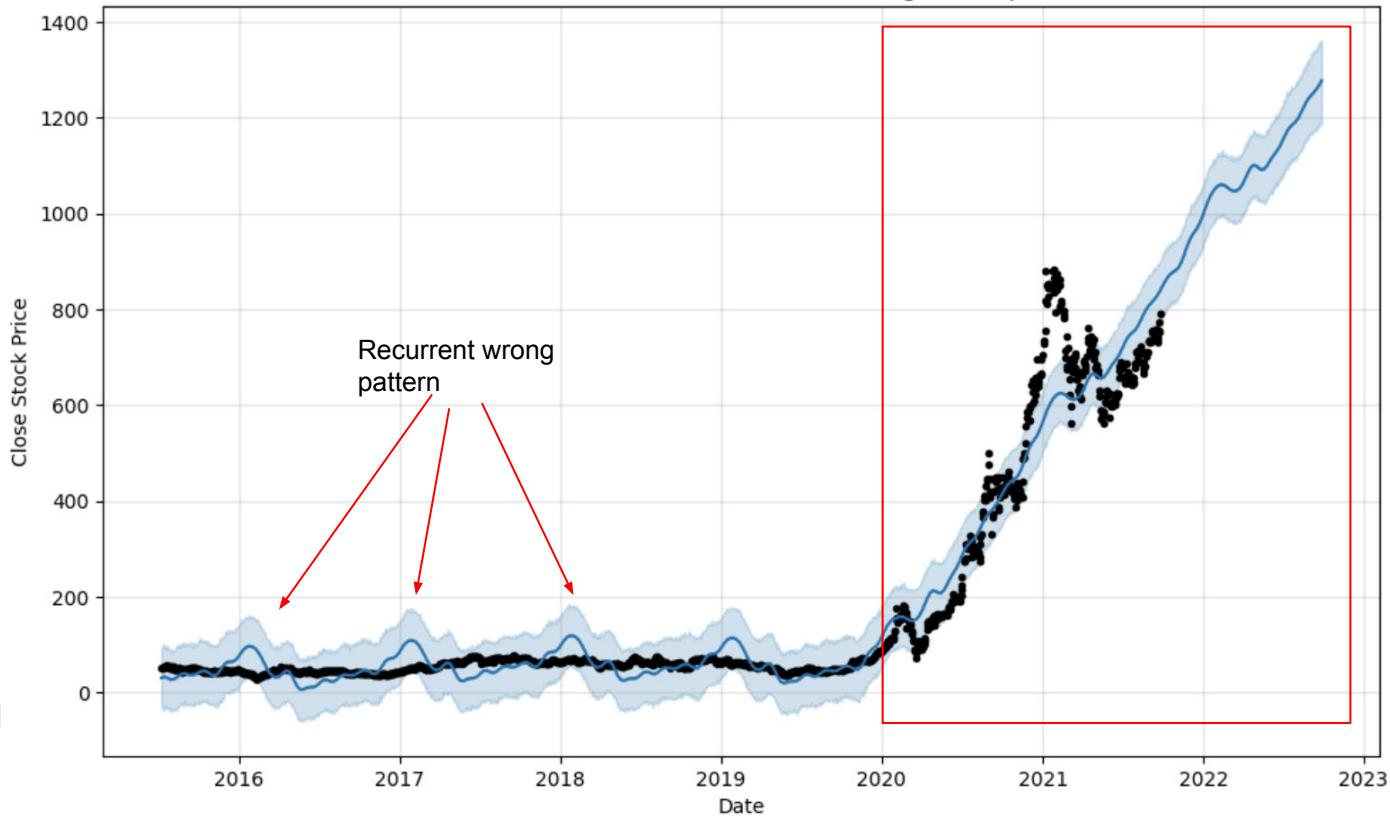
Prediction of the TESLA Stock Prices using the Prophet



¹ <https://facebook.github.io/prophet/>

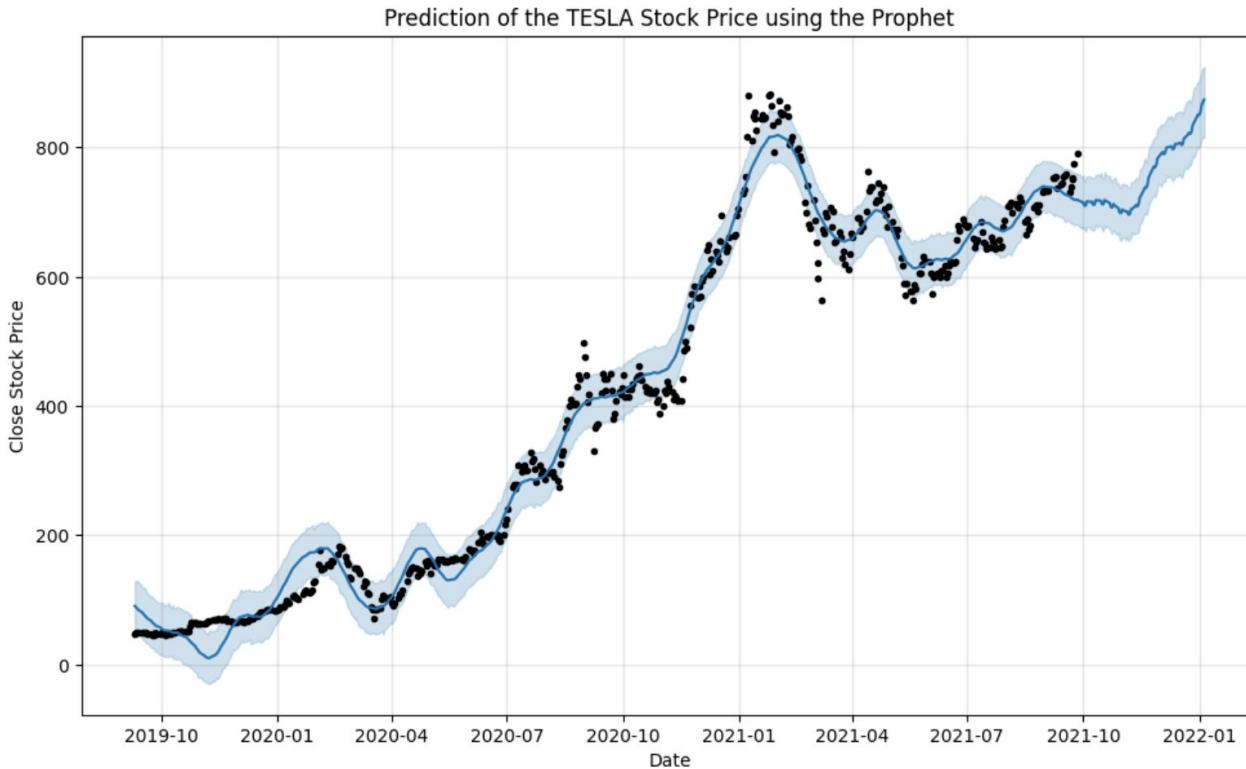
Application - Results

Prediction of the TESLA Stock Prices using the Prophet



¹ <https://facebook.github.io/prophet/>

Application - Results



Reducing the history the model could focus on the extreme part

¹ <https://facebook.github.io/prophet/>

Discussion/Conclusion

The results show that model **complexity does not improve performance**.

Training a deep learning model is not simple because it requires **hyperparameters optimization**.

The choice of the **window size** has a strong impact on the performance of the models.

The **data history** must be chosen with relevance.

Do not forget the more classical statistical models like ARMA, ARIMA, SARIMA...

It is important to always take a **baseline**.

Memory models (LSTM, GRU) remain the most efficient in single layer or bidirectional (only considering Deep Learning).

Stock prices are **hard to predict**, or **even impossible** if the market fluctuates too sharply.

Conclusion



Future reading

Lara-Benitez & Carranza-Garcia & Riquelme, 2021. *An Experimental Review on Deep Learning Architectures for Time Series Forecasting*. ArXiv.

<https://arxiv.org/pdf/2103.12057.pdf>

→ Comparison between Deep Learning and statistical approaches

Future

- Self-Supervised learning
- Quantum Machine Learning

Github

You can access to the notebook, data and presentation here:

- <https://github.com/Christophe-pere/DSDT-timeseries-rnn>

Merci / Thank You

Data Science | Design | Technology

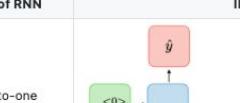
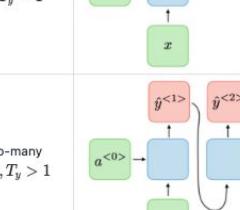
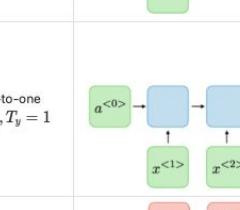
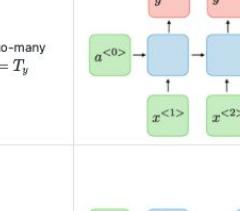
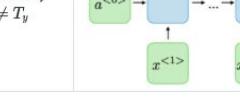
(Check for next DSDT meetup at [meetup.com/DSDTmtl/](https://www.meetup.com/DSDTmtl/))

@DsdtMtl



<http://bit.ly/dsdtmtl-in>

RNN Family - Architectures

Type of RNN	Illustration	Example
One-to-one $T_x = T_y = 1$		Traditional neural network
One-to-many $T_x = 1, T_y > 1$		Music generation
Many-to-one $T_x > 1, T_y = 1$		Sentiment classification
Many-to-many $T_x = T_y$		Name entity recognition
Many-to-many $T_x \neq T_y$		Machine translation