Comparison Study of Models for Time Series Forecasting

HSO

January 18, 2021

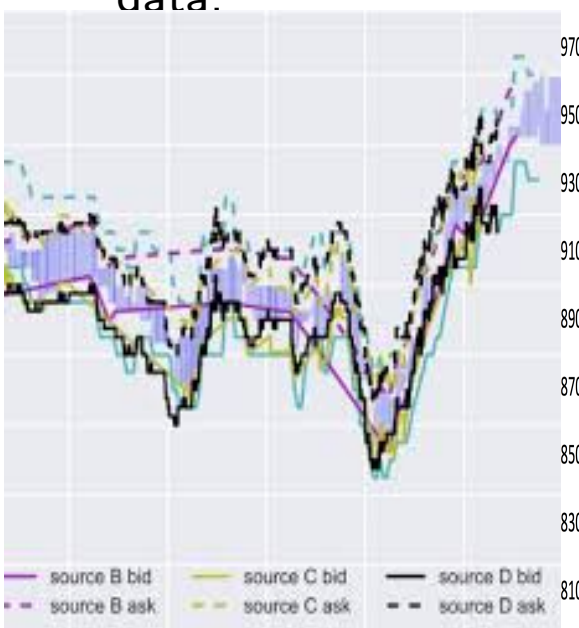
Outline

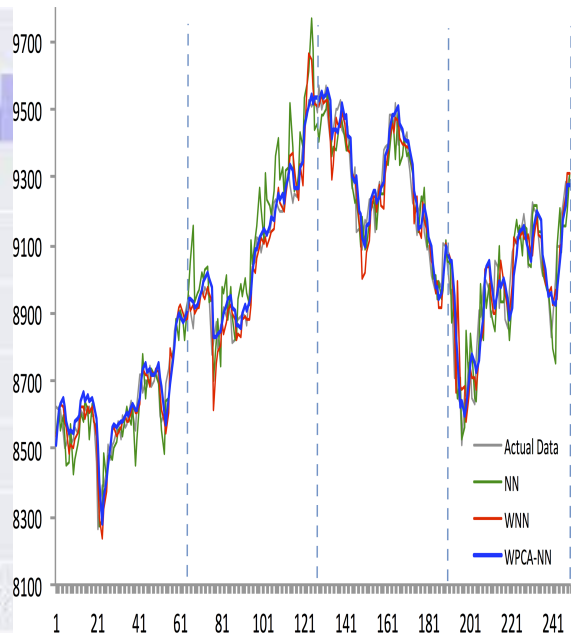
1. Contextualization

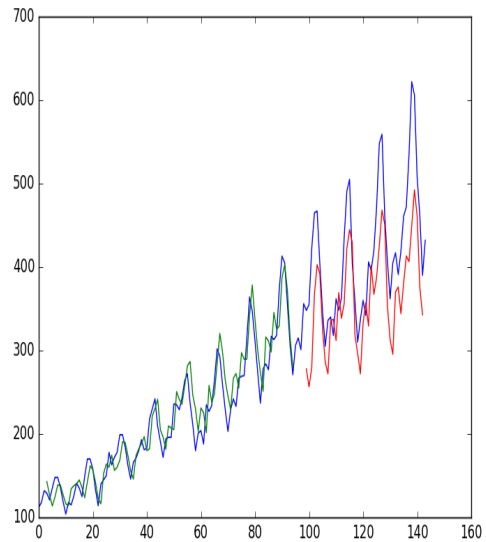
- Time Series are present in many important fields of our society.
- Economics, Medicine, Electronics.

2. Time Series

- Time series forecasting is area of machine learning that is often neglected.
- Real problems commonly involve a time component.
- The time component makes time series problems more difficult to handle.
- In Time Series Forecasting, we make predictions about the future.
- Forecasting involves taking models fit on historical data and using them to predict future observations.
- In forecasting the future is unavailable and is only estimated from the past.
- Contrary, descriptive models use all data and are used to describe the data.

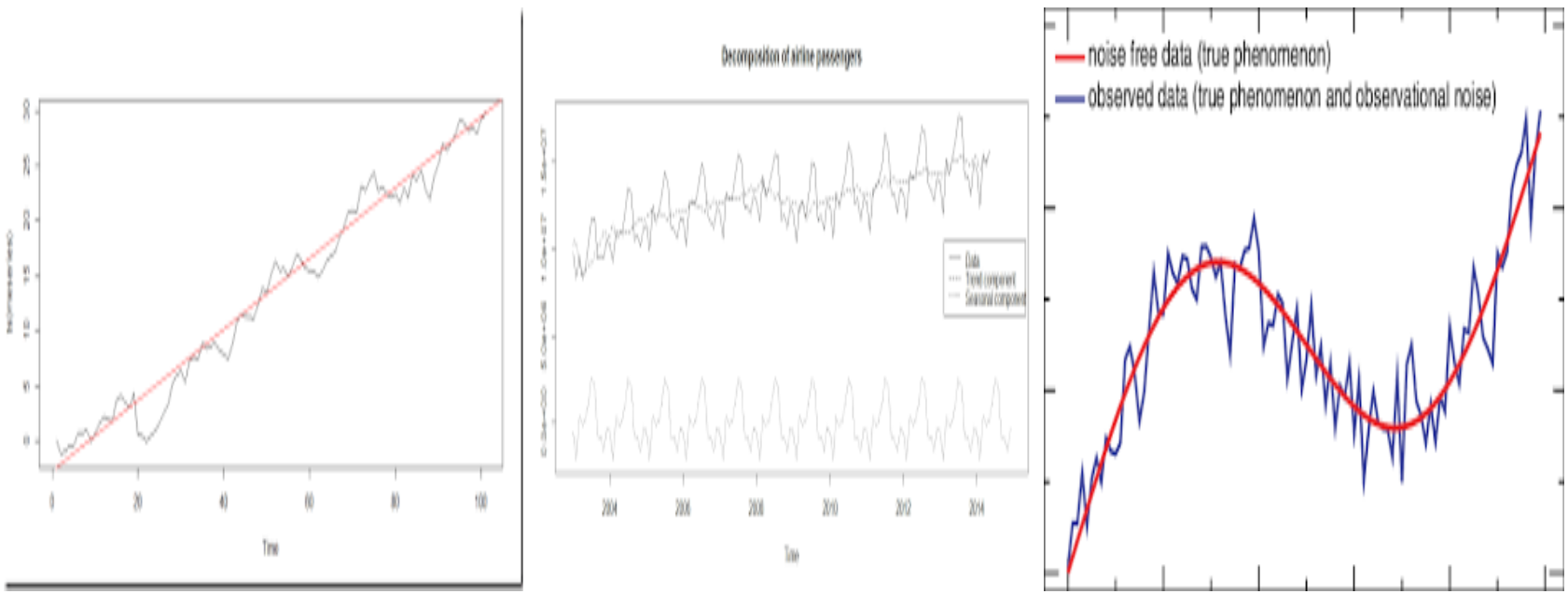






2. Time Series Characteristics

- **Level** - The baseline value for the series if it were a straight line.
- **Trend** - Linear increasing or decreasing behavior of the series over time.
- **Seasonality** - Repeating patterns or cycles of behavior over time.
- **Noise** - Variability in the observations that cannot be explained by the model (Residuals).



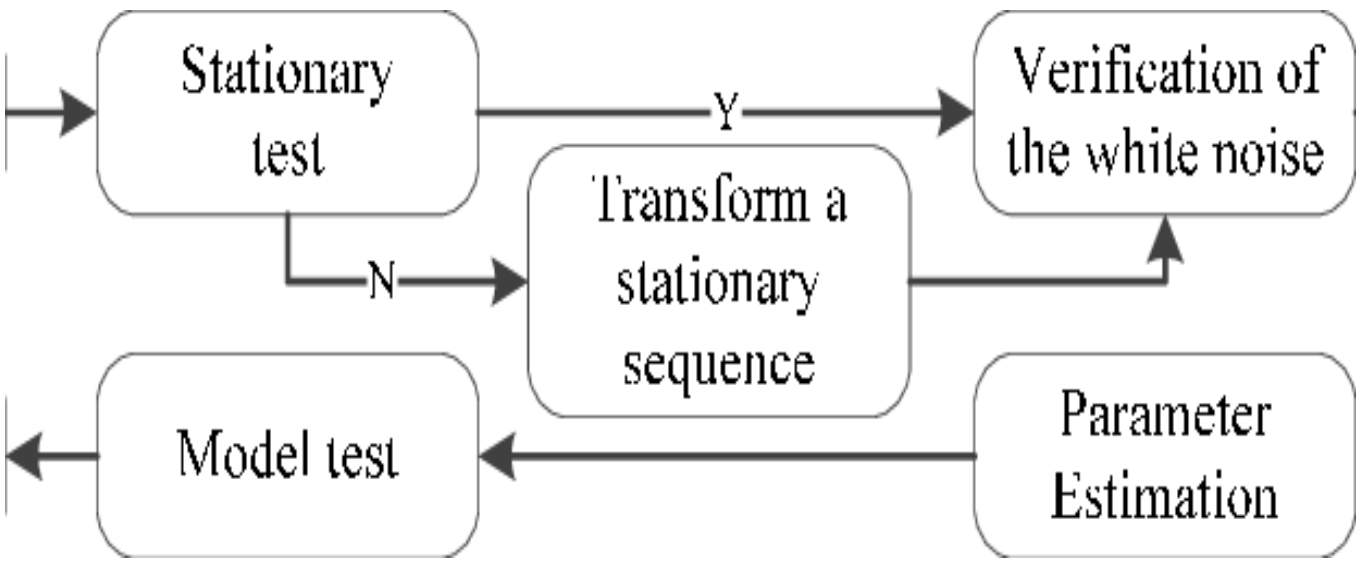
Example of different time series Characteristics

3. Background - ARIMA

Auto-Regressive Itegrative Moving Average (ARIMA), also Box-Jenkins model, corresponds to a generalization of the Auto-Regressive Moving Average (ARMA) model by including an integrative component I.

These integrated components are useful when data is non-stationary, and the integrated part of ARIMAs contributes to reducing the non-stationary.

The ARIMA applies differentiation on time series one or more times to remove the non-stationary effect. The $ARIMA(p, d, q)$ represents the order of Auto Regression (AR), and Moving Average (MA), and differentiation components.



Arima Flow Chart

3. Background - LSTM

Recurrent Neural Nets (RNN) has some difficulties learning long-range dependencies.

Long Short Term Memory (LSTM) addresses the problem of vanishing gradients.

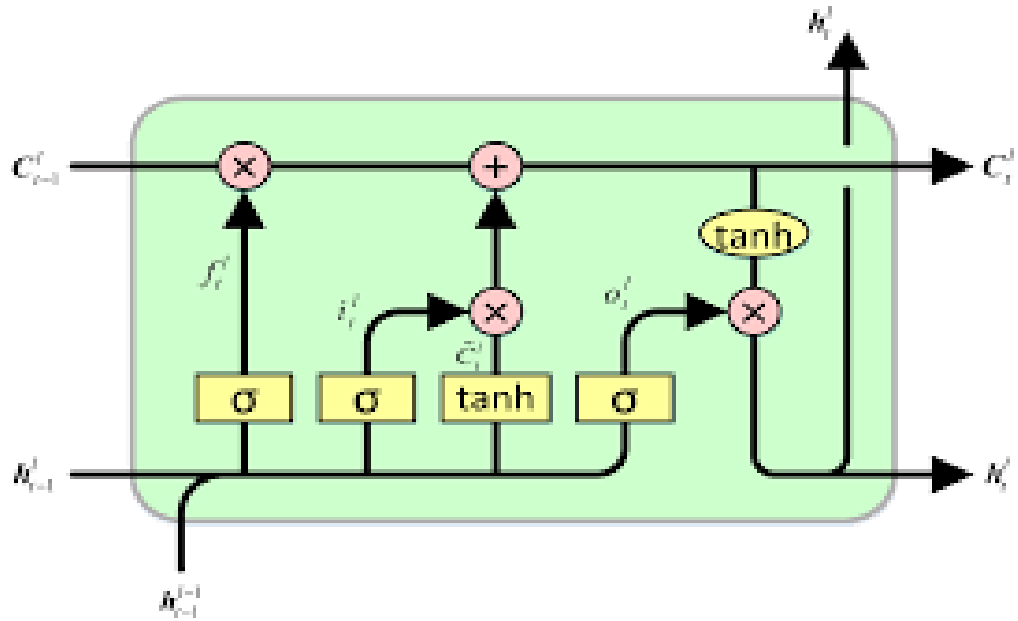
BackPropagation Thought Time (BPTT) is a variant of the back-propagation algorithm.

LSTM introduces an additional computations on each time-step.

For time-step h_t, returns the state internal (f_t) and this is forwarded to the next time-step.

LSTM introduces three new gates: The input (o_t), forget g_t and output c_t gates.

$$o = \sigma(W^o x_t + U^o s_{t-1})$$
$$g_t = \tanh(W^g x_t + U^g s_{t-1})$$
$$c_t = f_t \odot c_{t-1} + h_t \odot g_t$$
$$s_t = \tanh(c_t) \odot o_t$$
$$h_t = f_t$$

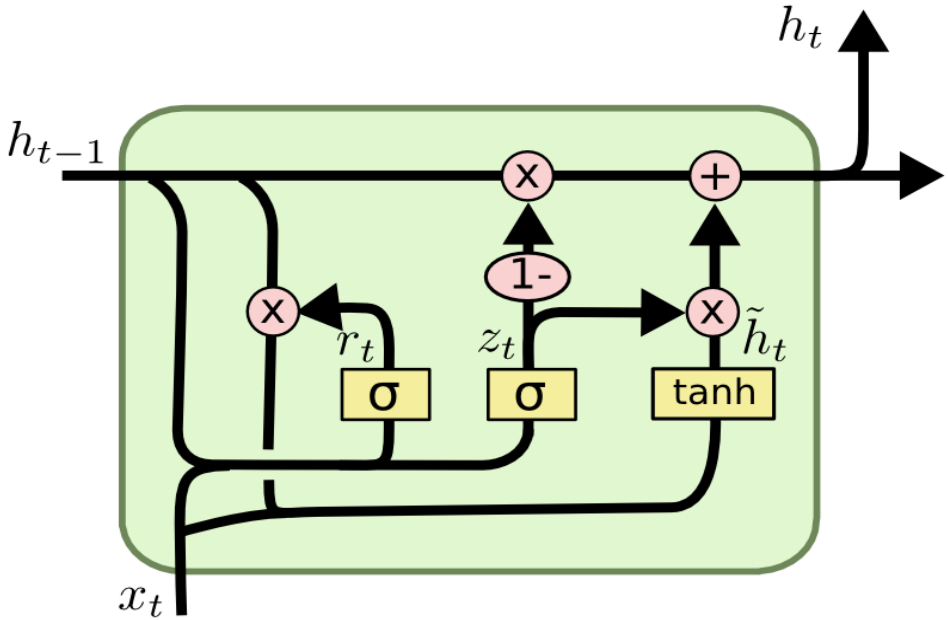


LSTM main architecture

3. Background - GRU

Gated Recurrent Unit (GRU) are derived from LSTM and are a simpler form that only has two internal states, namely, the update gate z_t and the reset gate r_t. The computations of the update and reset gates are:

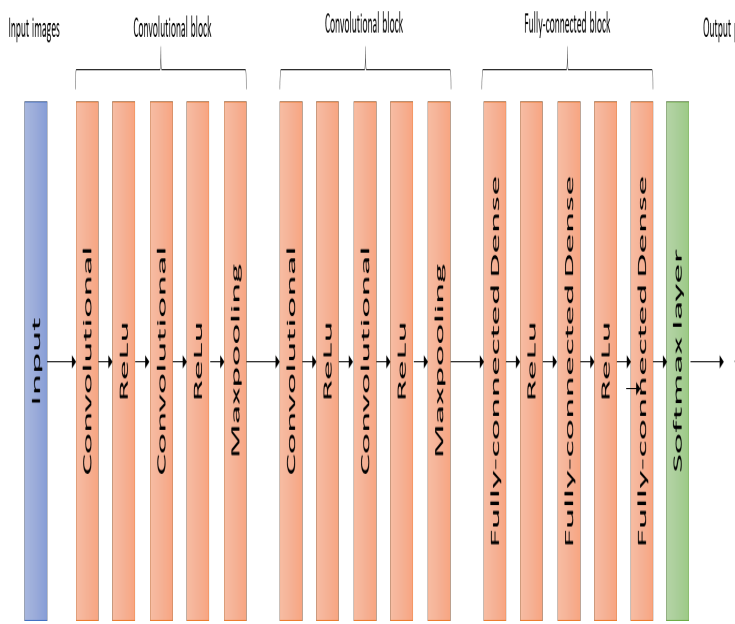
$$r_t = \sigma(W^z x_t + U^z s_{t-1})$$
$$s_t = z_t \odot s_{t-1} + (1 - z_t) \odot \tanh\left((W^h x_t + U^h(r_t \odot s_{t-1}))\right)$$



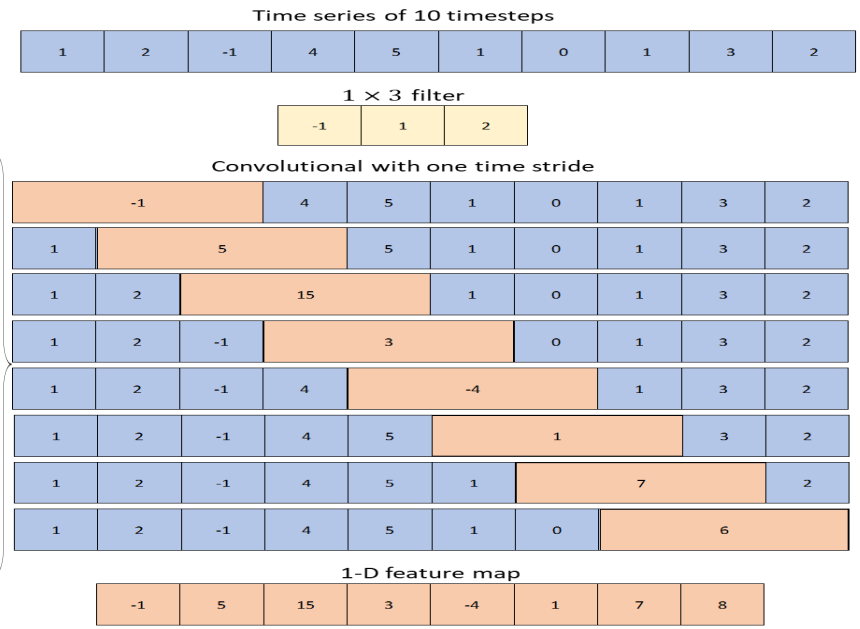
GRU cell main architecture

3. Background - CNN

Convolutional Neural Networks (CNN) have been widely used to address time series problems with great success.
The approach of using a 1×3 convolution filter corresponds to training several local autoregressive models of order.
Several 1D convolutions and pooling layers, when stacked with each other, give a powerful way of extracting features from the original time series



CNN main architecture

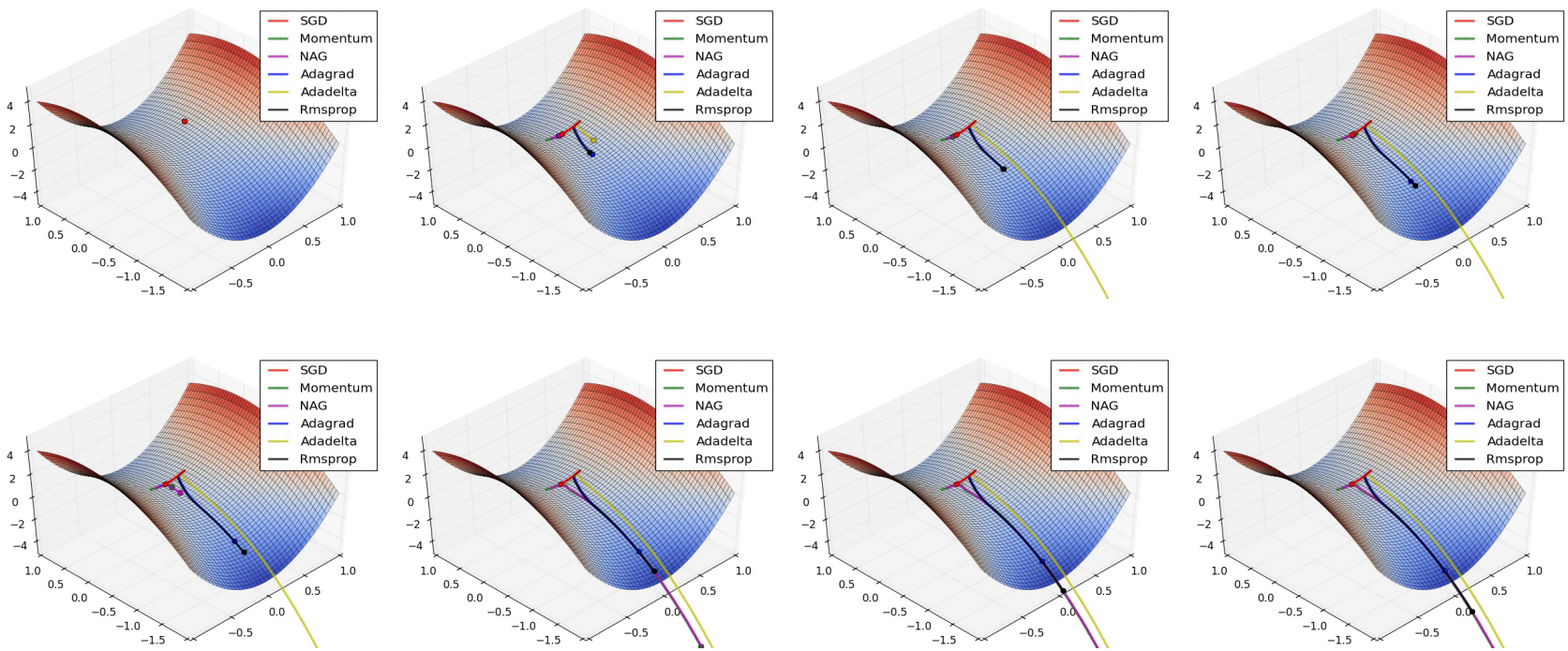


Convolution filtering

4. Optimizer

Selection of the optimizer is a hyper-parameter tuning task. Many optimizer exist each one with limitations and advantages:

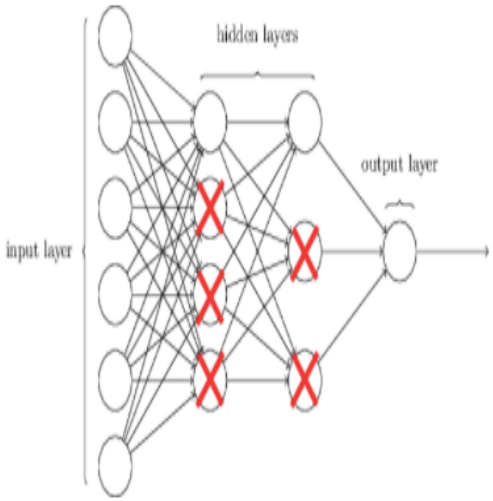
1. AdaGrad penalizes the learning rate too harshly.
2. AdaDelta,RMSProp almost works on similar. Adadelta don't require an initial learning rate.
3. Adam combines the good properties of Adadelta and RMSprop.
4. SGD is very basic and is seldom used now. Has a hard time escaping the saddle points.

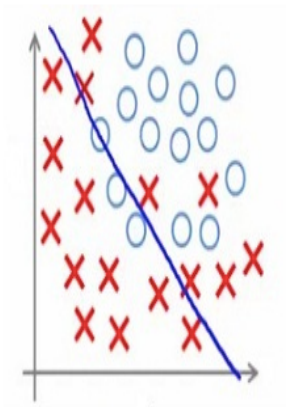


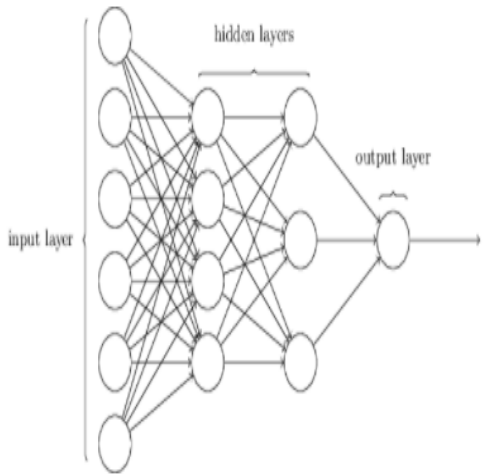
4. Regularization

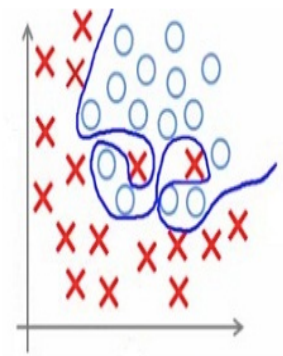
One of the most common problem data science professionals face is to avoid overfitting.

Avoiding overfitting can single handedly improve our model's performance.

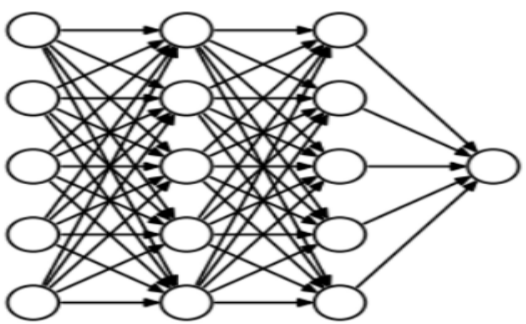




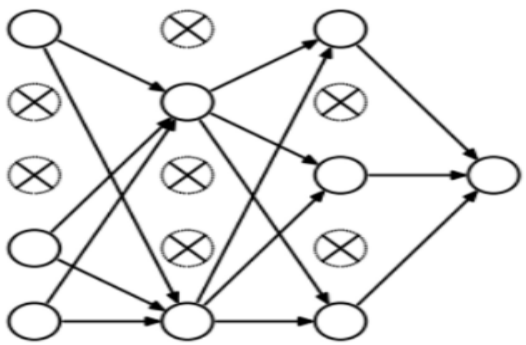




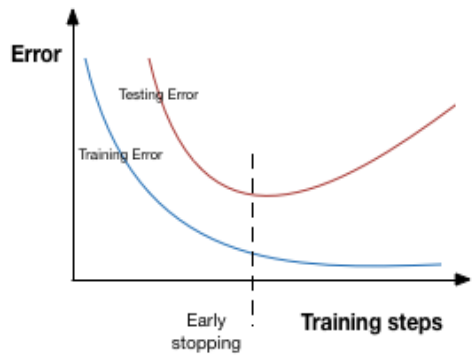
L2 and L1 regularization $CostFunction = Loss + \frac{\lambda}{2m} \times \sum ||w||^n$



FC



FC with Dropout



Early stop

5. Evaluated Datasets

Experiments in Uni variate and multivariate datasets to asses the performance of the models.
Side side comparsion using same datsets and experiments.
Briefly discuss each of the models in term of performance on the respective datasets.
Final Model using different dataset to access the generalization capabilities in multi variate time series.

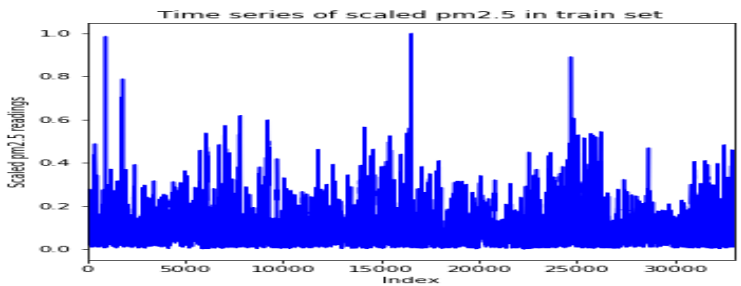
- Beijing PM2.5 Data Data Set - https://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data
- IBM common stock closing prices Data Data Set - https://www.nasdaq.com/market-activity/stocks/ibm/historical
- Google Stock Price - https://finance.yahoo.com/quote/GOOG/history/
- Tesla Stock Price - https://finance.yahoo.com/quote/TESLA/history/

Observations

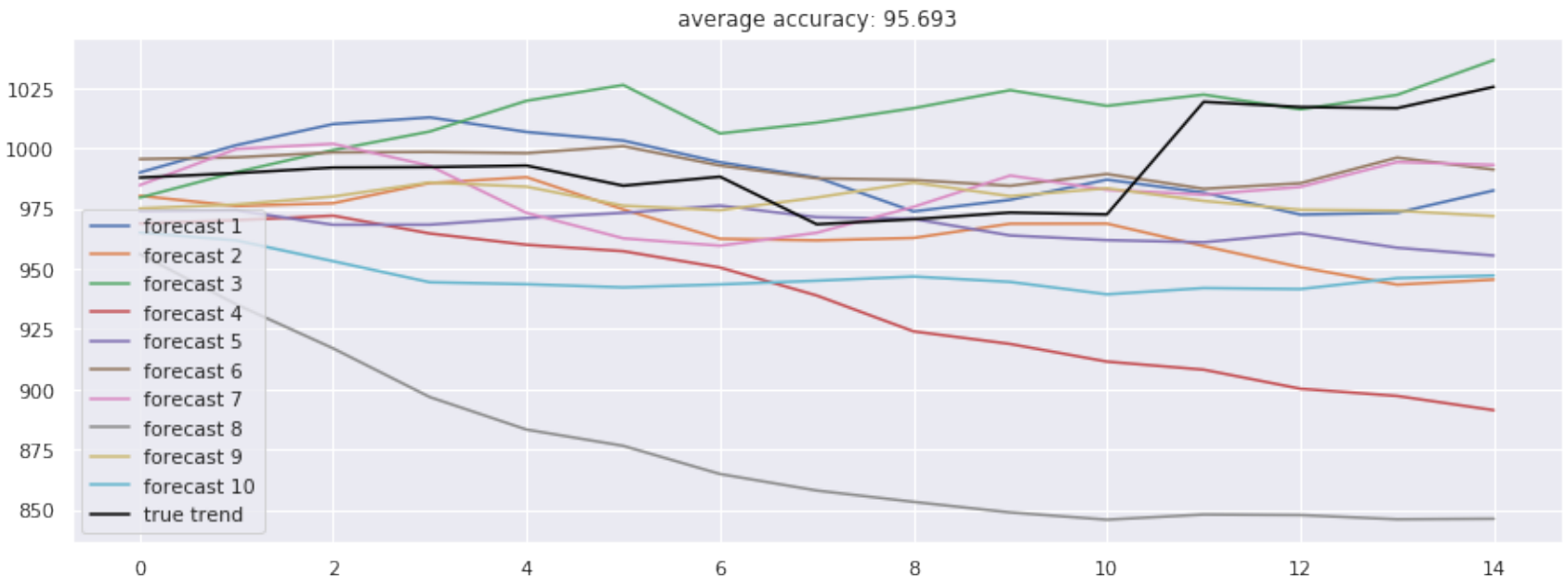
* The train dataset IS derived from starting timestamp until last 30 days
* Test dataset contains the last 30 days for forecasting
* Numerical features are normalized for training and the inverse transformation is performed to obtain the original numeric scale.

6. Models Evaluation - LSTM

Using the same dataset, we evaluate the model performance using RNN models.
Normalize the values between $[0 - 1]$ interval.
Adam optimizer $\eta = 0.001$ and decay factor of $1e^{-5}$. The batch size was set to 16, and trained 200 epochs.




```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 7, 1)              0
_________________________________________________________________
lstm_1 (LSTM)                (None, 7, 64)             16896
_________________________________________________________________
lstm_2 (LSTM)                (None, 32)                12416
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 33
=================================================================
Total params: 29,345
Trainable params: 29,345
Non-trainable params: 0
_________________________________________________________________
```

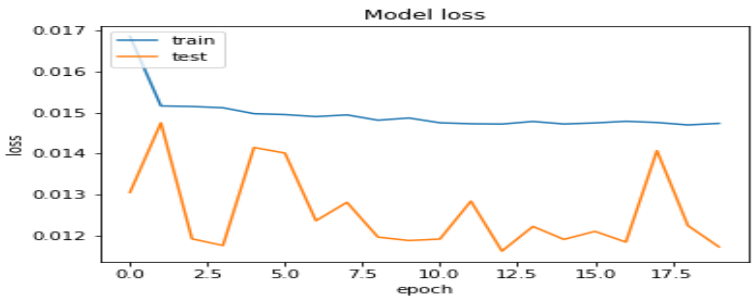


LSTM Actual vs Predicted

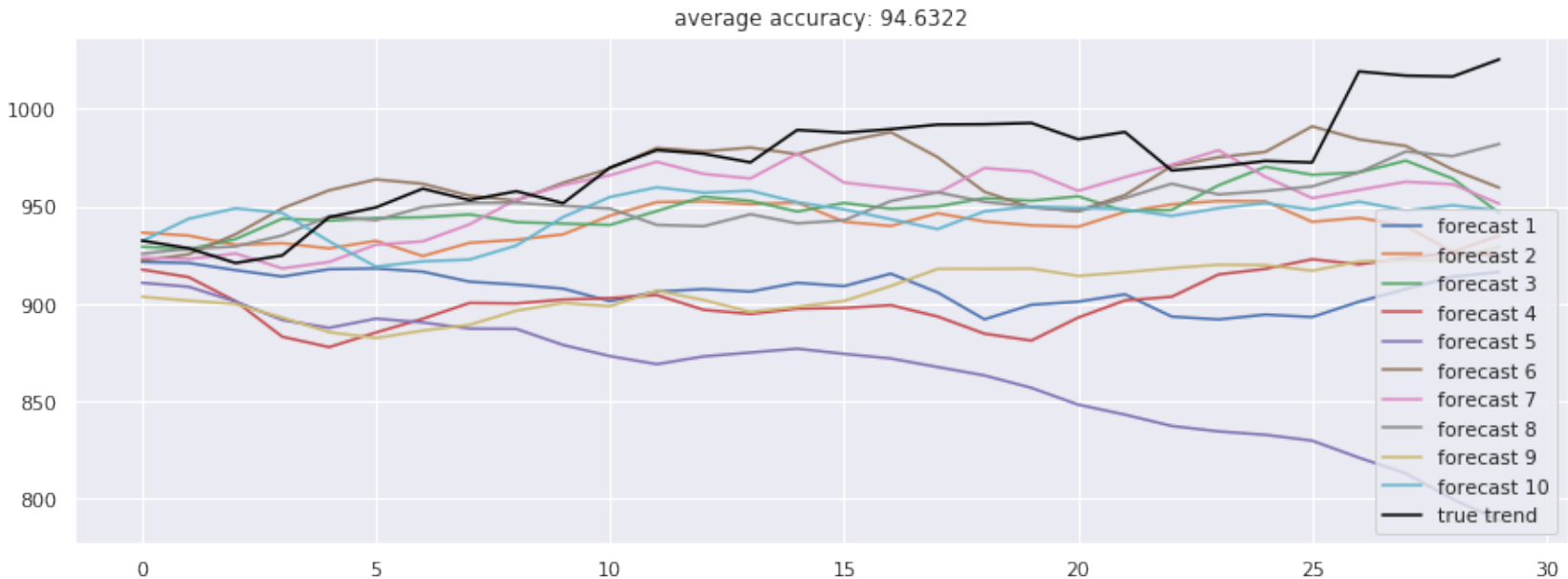
6. Models Evaluation - GRU

GRU layers and a Dropout layer and dense layer.

```
Model: "model_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 7, 1)              0
_________________________________________________________________
gru_1 (GRU)                  (None, 7, 64)             12672
_________________________________________________________________
gru_2 (GRU)                  (None, 32)                9312
_________________________________________________________________
dropout_2 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 22,017
Trainable params: 22,017
Non-trainable params: 0
_________________________________________________________________
```



GRU model architecture



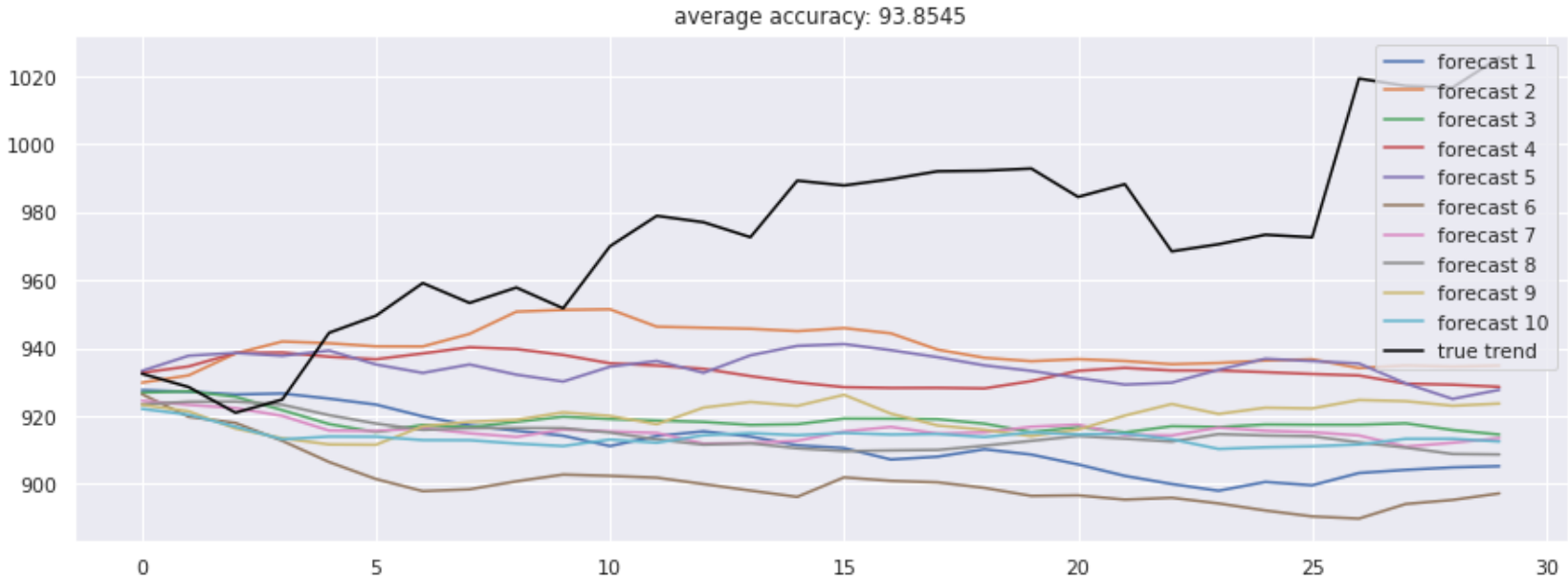
GRU Actual vs Predicted

6. Models Evaluation - LSTM + Bi-Directional

Unidirectional LSTM only preserves information of the past because the only inputs it has seen are from the past.

Bi-Directional will run the inputs in two ways, one from past to future and vice-versa.

Improves Generalization



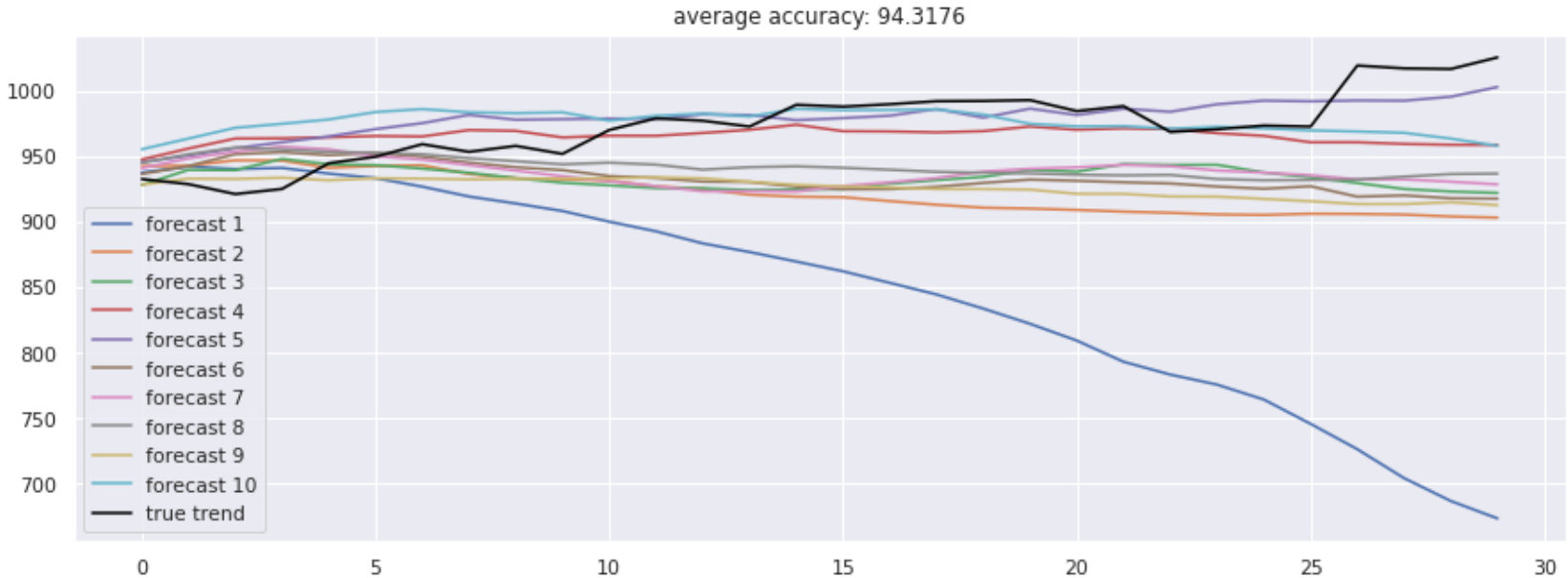
LSTM Bidirectional Training Actual vs Predicted

Observations

We see a little downgrade in the performance when compared with LSTM, however for longer sequences is possible that the model is able to generalize better.

6. Models Evaluation - LSTM + Seq2Seq

A sequence to sequence model maps a fixed-length input with a fixed-length output where the length of the input and output may differ.
Encoder formed by a stack of several recurrent units LSTM
The Decoder, a stack of several recurrent units to predicts y_t at a time step t.
Suitable for time series forecasting of any length.



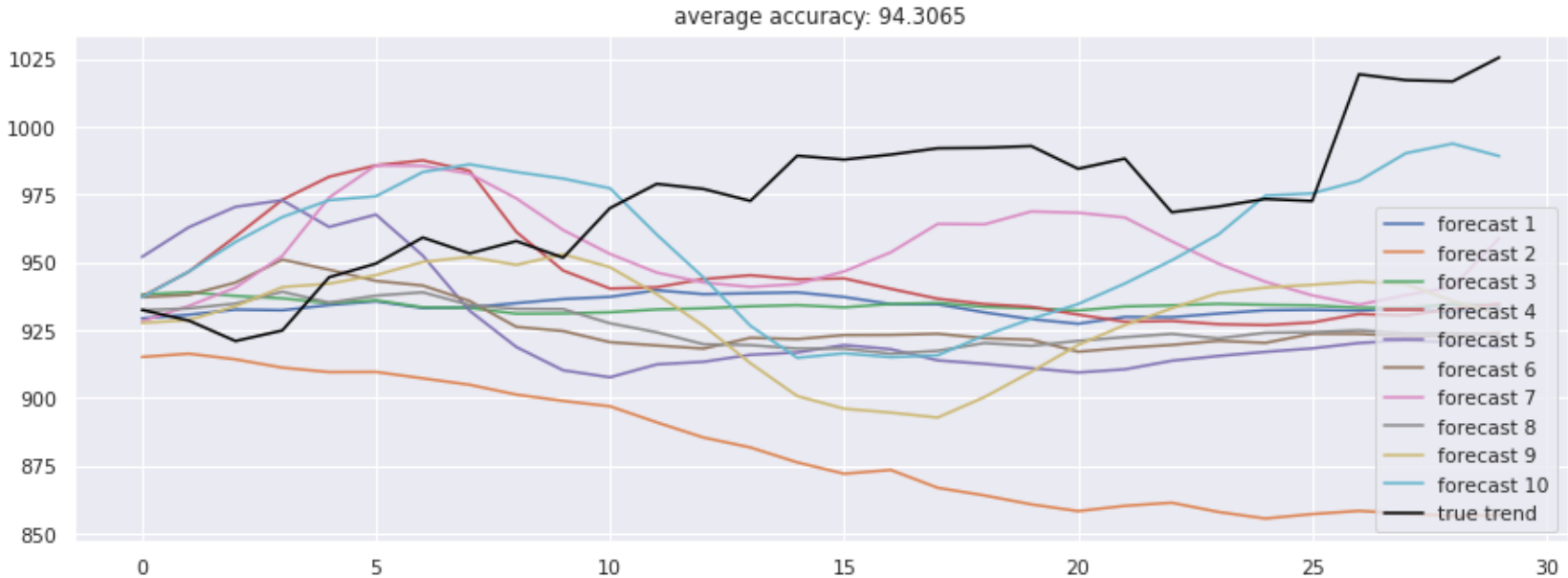
LSTM Sequence 2 Sequence Actual vs Predicted

Observations

The extra complexity of the encoder-decoder configuration in this case does not add significant improvements to model performance.

6. Models Evaluation - LSTM + Seq2Seq + Bi-Directional

Re-usage the last model, but now but making a 2 Bi-Directional LSTM.
Bidirectional enables to capture past and future information for forecasting.



LSTM Sequence 2 Sequence Bidirectional Actual vs Predicted

Observations

Results are compare with simple LSTM models.

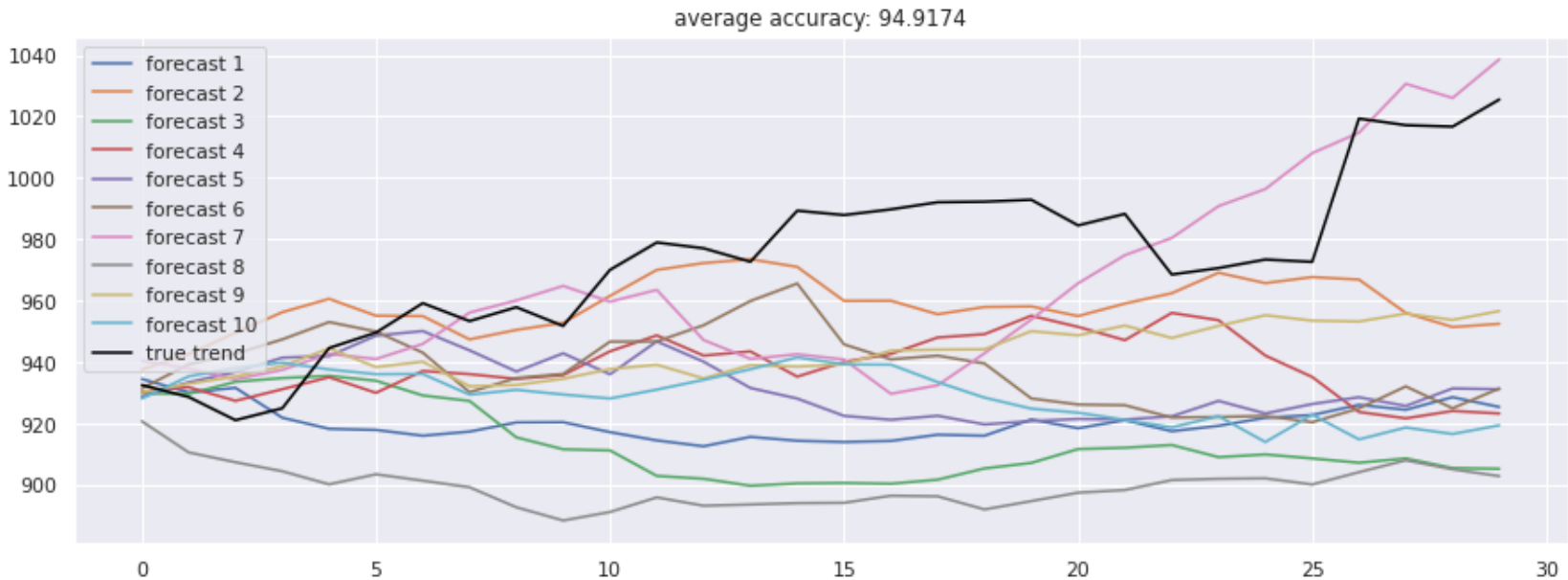
6. Models Evaluation - LSTM + Seq2Seq + VAE

Variational Auto Encoder (VAE) enforces an bottleneck

A VAE is a autoencoder avoids overfitting and ensure that the latent space has good properties.

A VAE composed of both an encoder and a decoder, trained to minimize the reconstruction error between the encoded-decoded data and the initial data.

The regularisation of the latent space is made by encoding as a distribution over the latent space.



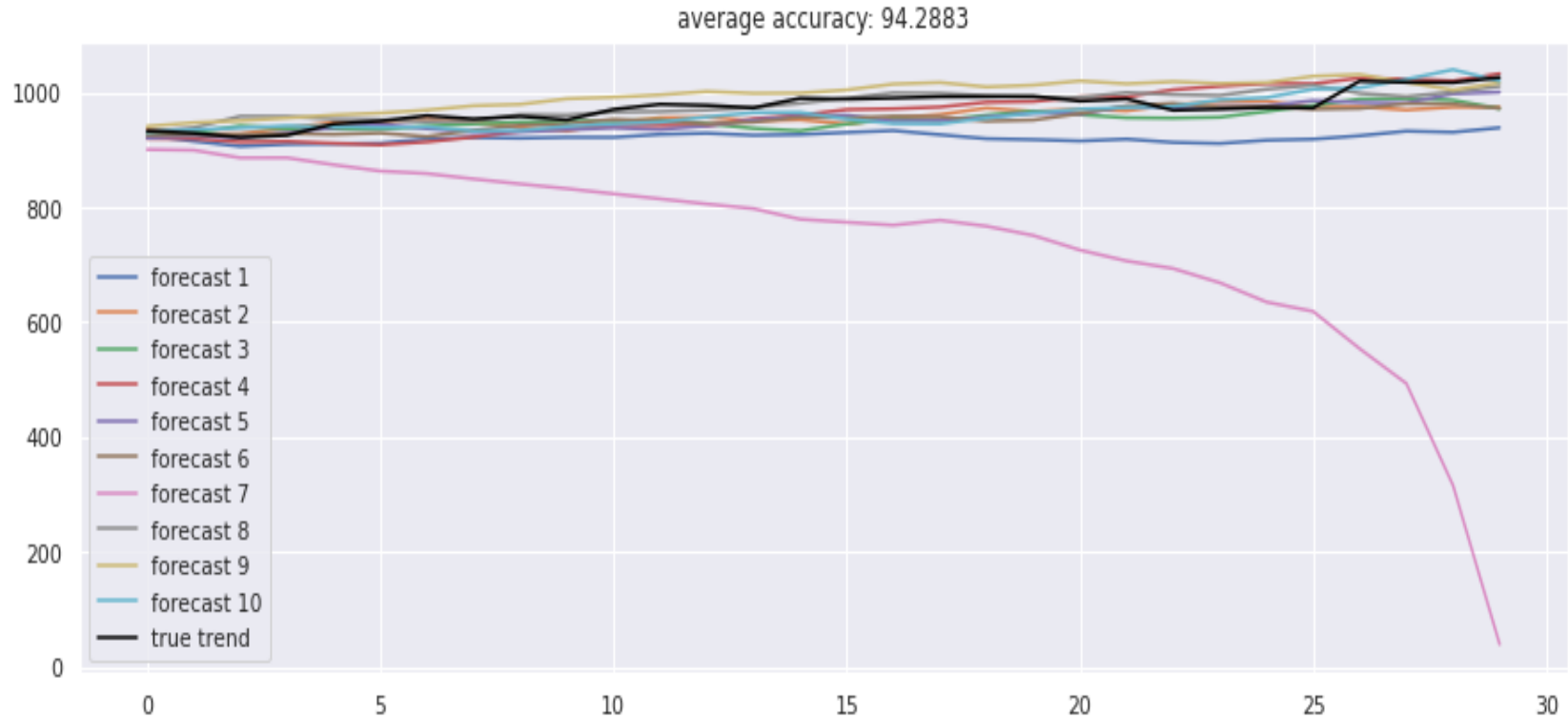
LSTM Sequence 2 Sequence VAE Actual vs Predicted

Observations

Improvement over encoder-Decoder models, suggesting that the bottleneck in fact avoid some degree of overfitting.

6. Models Evaluation - GRU + Bi-Directional

Similar to LSTM in its core, the GRU bidirectional also share common characteristics. The main difference is that GRU has fewer control gates when compared with LSTM.



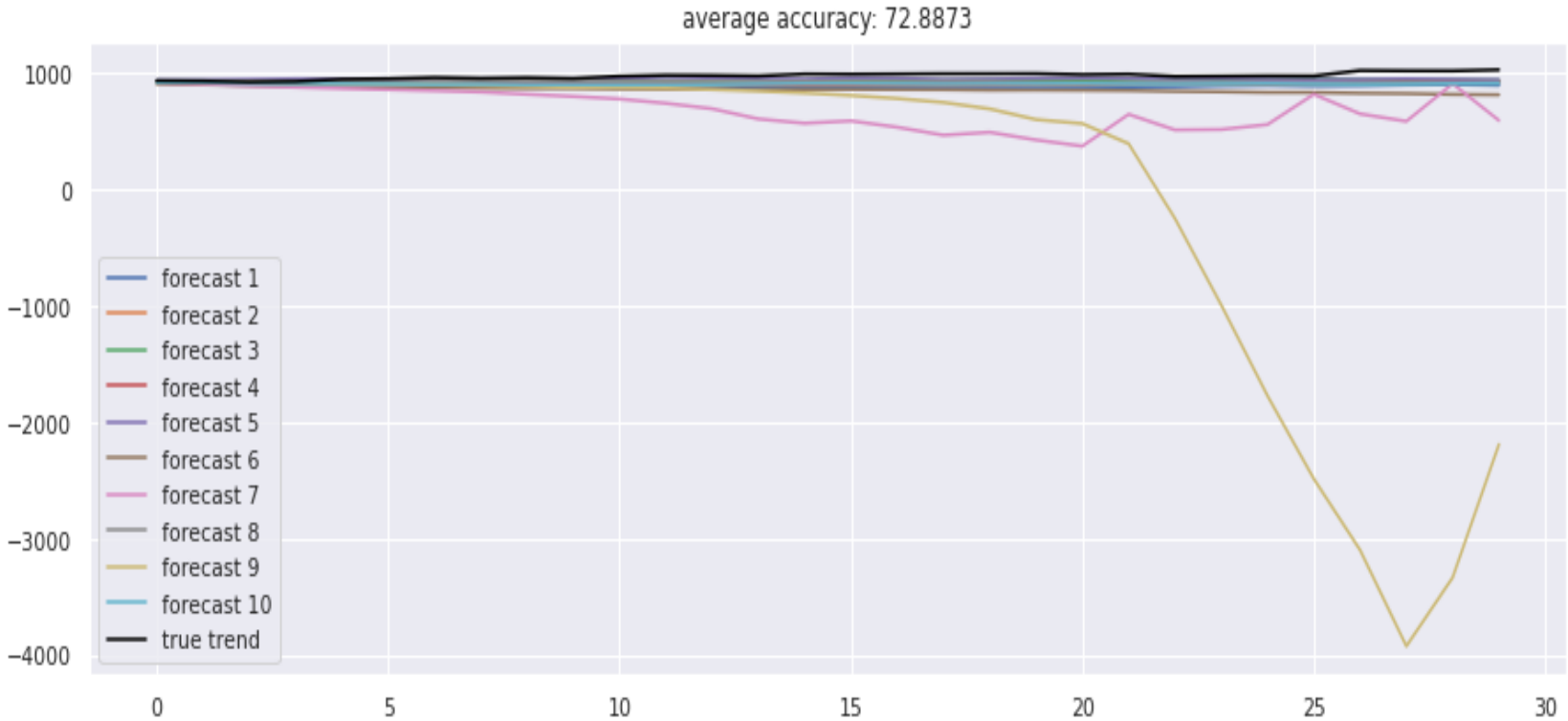
GRU Bidirectional Training Actual vs Predicted

Observations

Results are lower when compare with simple LSTM models, meaning that the extra complexity of bidirectional GRU configuration does not add significant improvements to model performance.

6. Models Evaluation - GRU + Seq2Seq

Similar conducted experiment of Encoder-Decoder, but now using GRU models.
ng



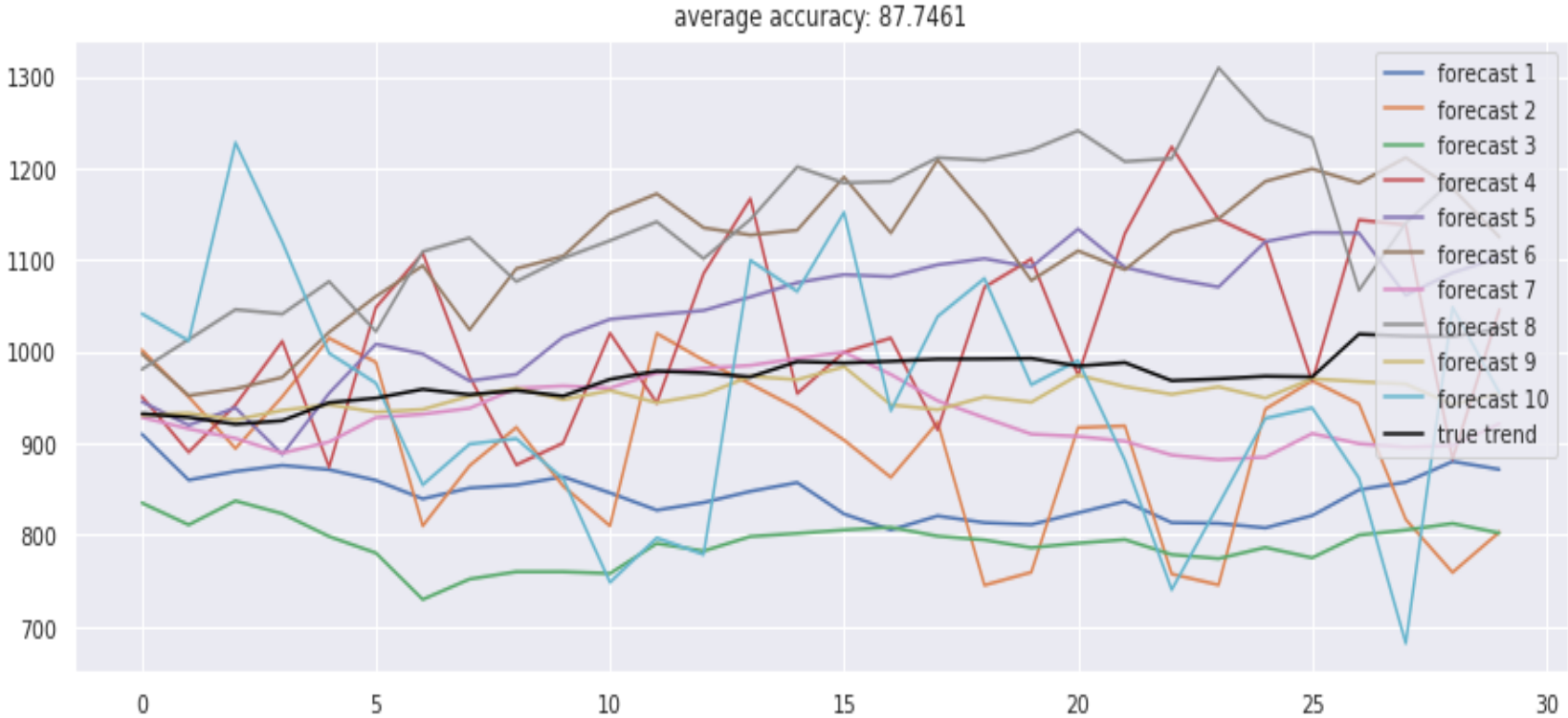
GRU Sequence 2 Sequence Actual vs Predicted

Observations

Results are lower when compare with simple LSTM models, meaning that the extra complexity of Encoder-Encoder in GRU configuration does not add significant improvements to model performance.

6. Models Evaluation - GRU + Seq2Seq + VAE

Ion this experiment, we explore the performance of Encoder-Decoder combined with a VAE to encapsulate the relevant information.



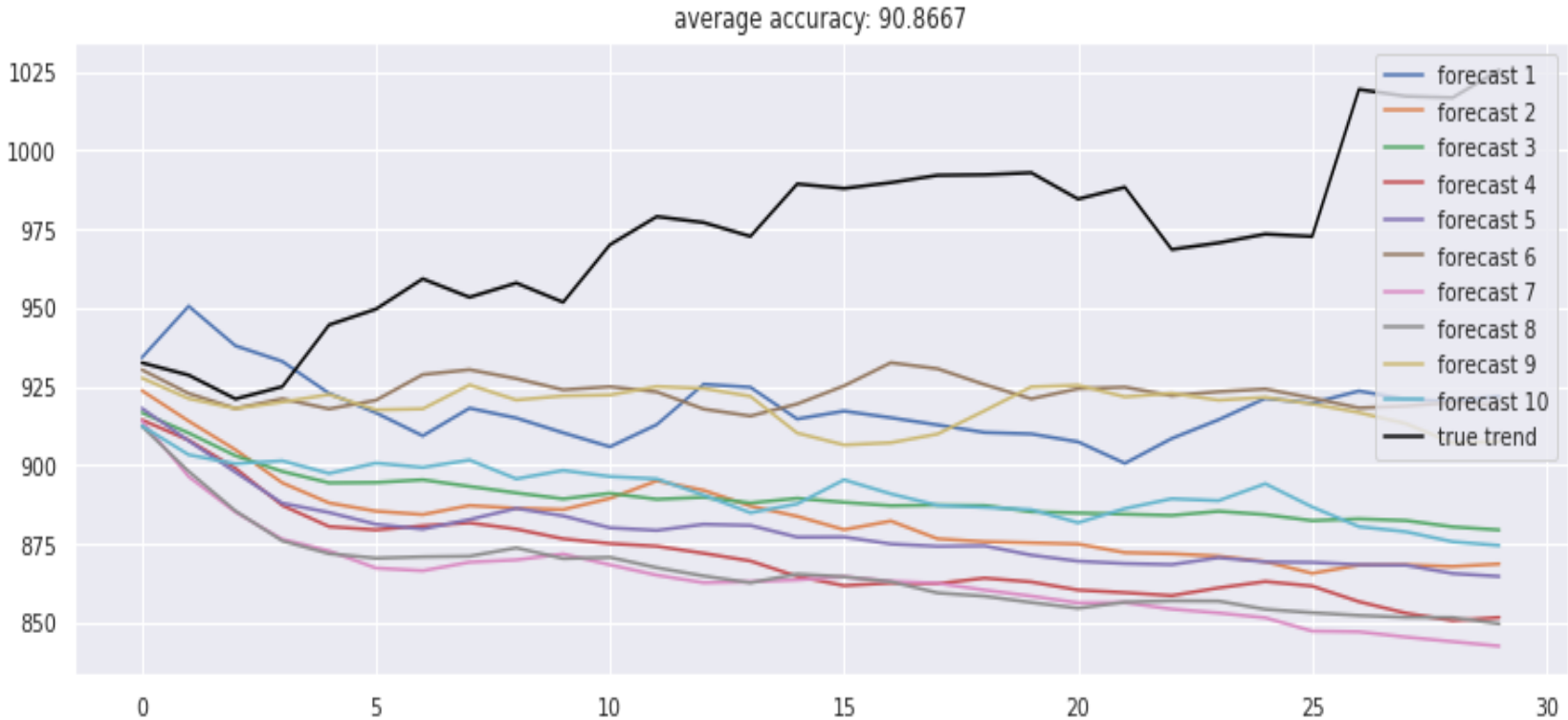
GRU Sequence 2 Sequence VAE Actual vs Predicted

Observations

Results are even lower than expected, with simple LSTM models, meaning that the extra complexity of bidirectional GRU configuration does not add significant improvements.

6. Models Evaluation - CNN + Seq2Seq

CNN in a Encoder-Decoder configuration.



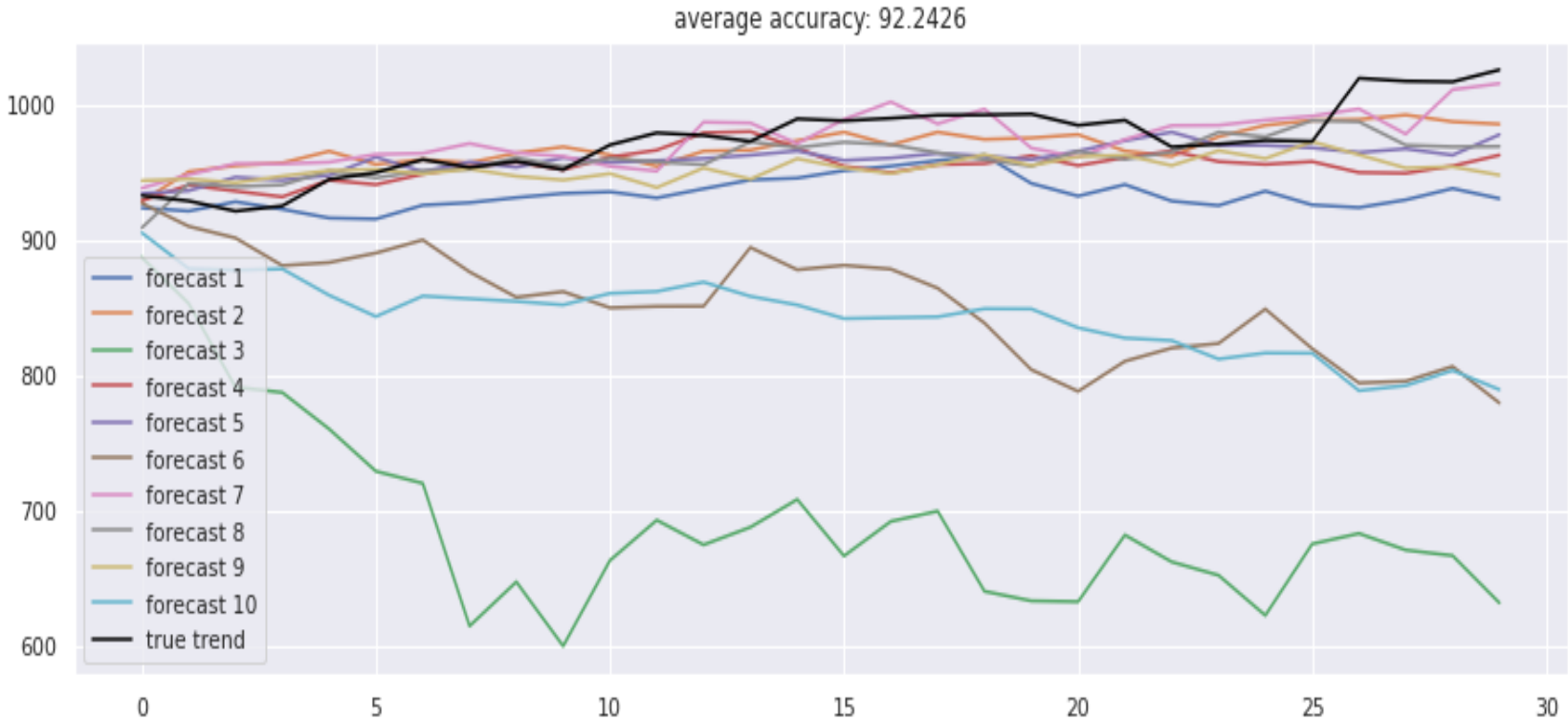
CNN Sequence 2 Sequence Actual vs Predicted

Observations

Results are lower than expected, meaning that the extra complexity of simple Encoder-Decoder in CNN configuration does not add significant improvements to model performance.

6. Models Evaluation - CNN + Dilated Seq2Seq

CNN in a Encoder-Decoder configuration using dilated convolutions.



CNN Dilated Sequence 2 Sequence Actual vs Predicted

Observations

Results are more robust, meaning that the increase of space time-space resolution in CNN configuration adds significant improvements to model performance.

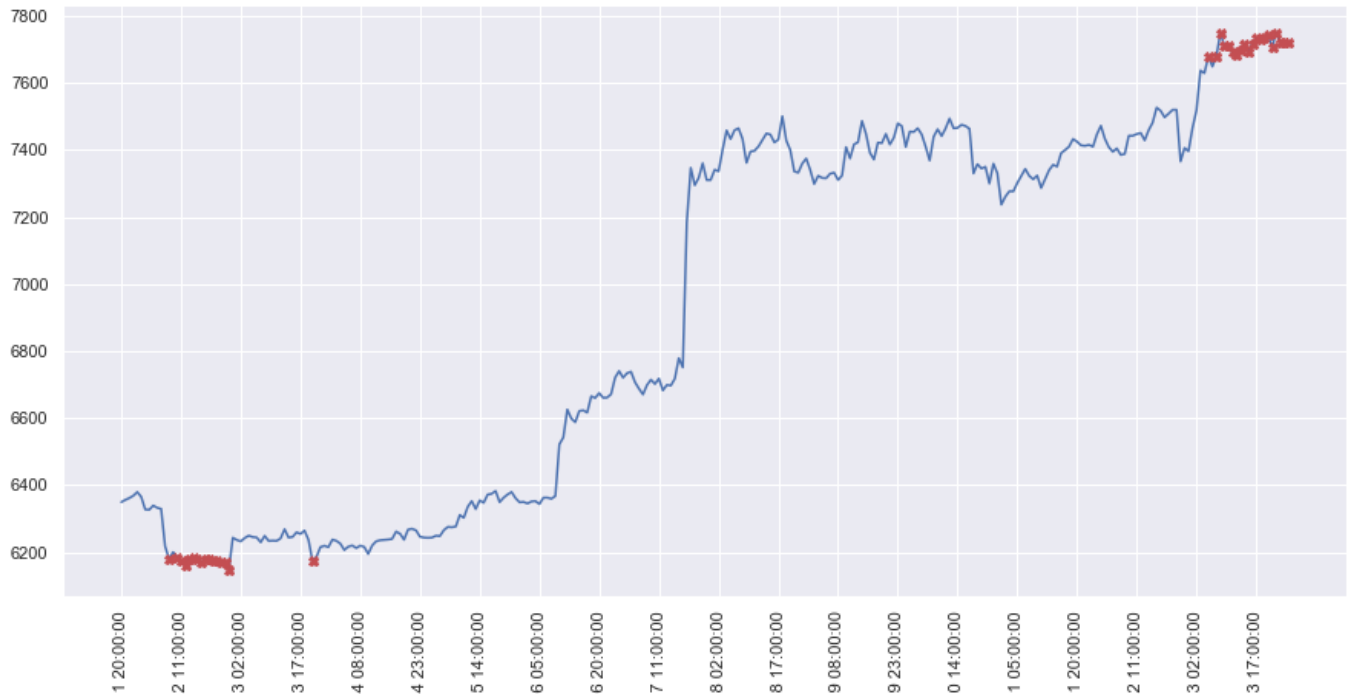
6. Models Evaluation - Stack ensemble with XGB

On this experiment, we explore the use of encoder feature combined with several regressors models. The encoder is responsible to encapsulate the relevant information of the training set on a latent reduced space, enabling for models to reuse the encoded feature information for model fitting. This approach is analogous to Principal Component Analysis (PCA) for feature dimensionality reduction, but in this case the obtained training daqta is encoded using a supervised Encoder-Decoder model optimized using RMS Prop. The evaluated models on the encoded training features are the AdaBoost, Bagging, Extra Trees Regressor, Gradient Boosting Regressor (GB), and Random Forest Regressor (RF). Result are summarized in Fig. **??**, and is possible observe that models fitted well to the training data.

6. Models Evaluation - LSTM for Multi Variable

As our final goal, we evaluate the performance of Multivariate Time Series (MTS) multi variate time series.

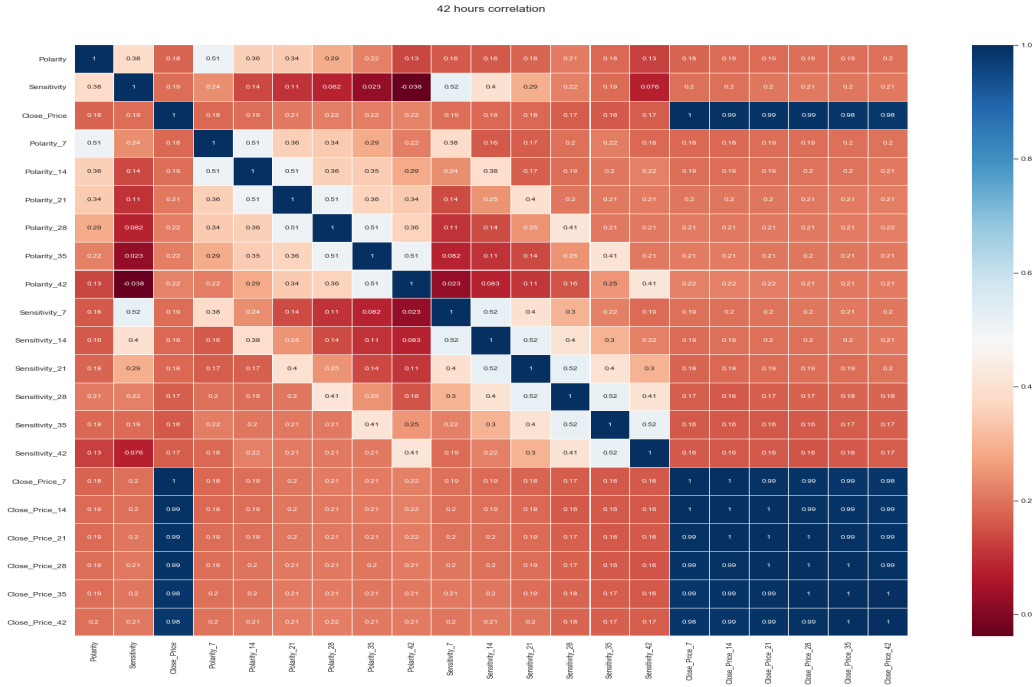
As summary of our experiments, we now present a experiments in a different dataset, containing multiple variable as predictors and a continuous target variable, the stock value.



MTS peaks and trends

Observations

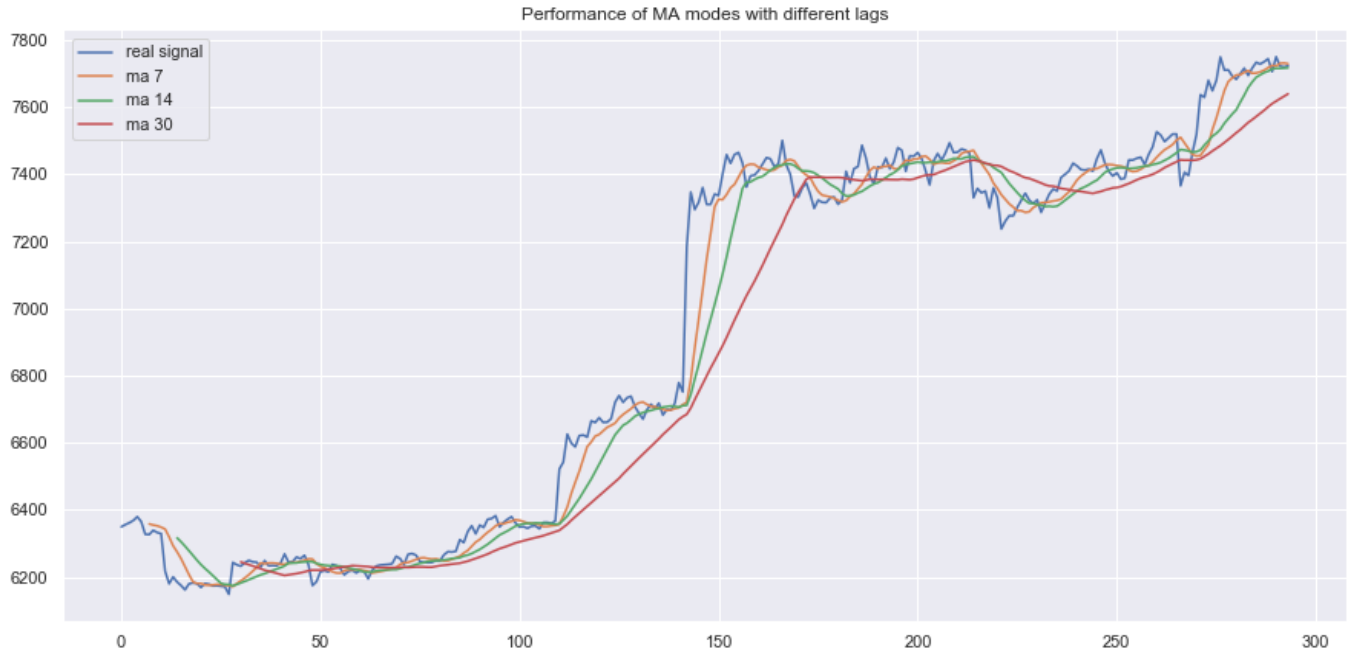
6. Models Evaluation - LSTM for Multi Variable



MTS 42 Hours Feature Pearson Correlation

Observations

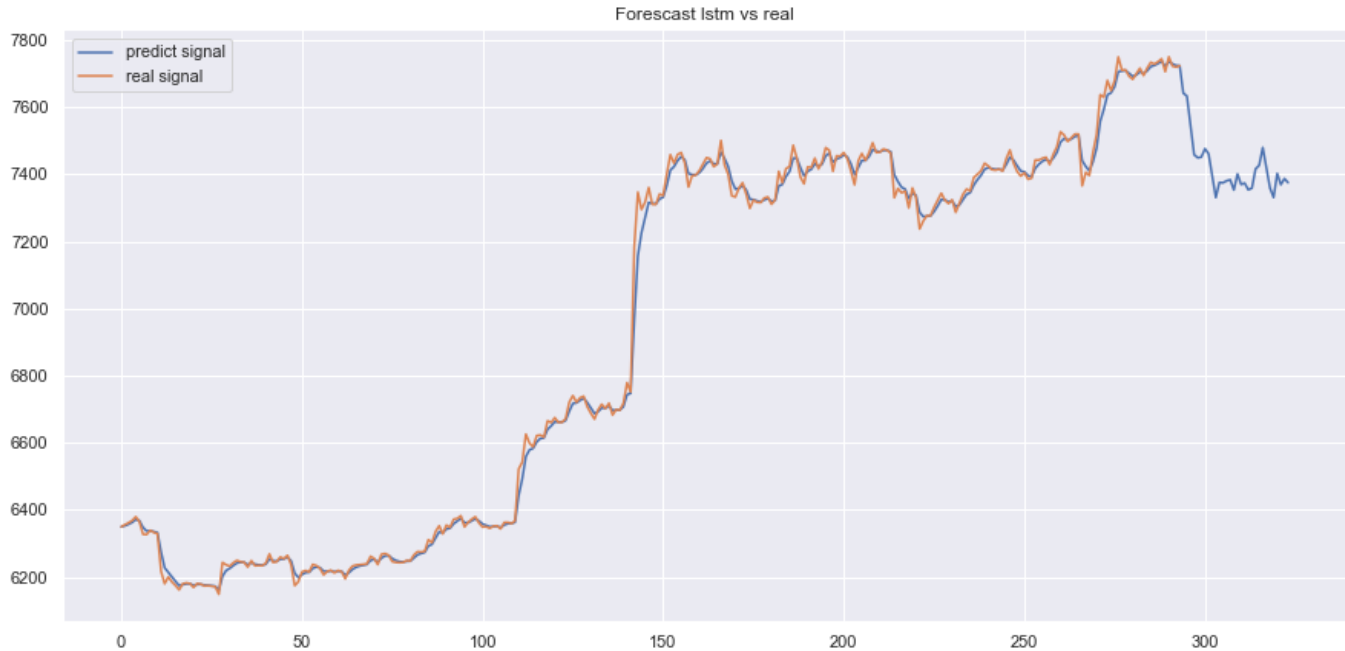
6. Models Evaluation - LSTM for Multi Variable



MTS Simple Moving Average evaluation with different lags

Observations

6. Models Evaluation - LSTM for Multi Variable



MTS Forecasts vs Ground Truth (GT)

Observations

7. Models Discussion

8. Conclusions

* Several models have theirs advantages and problems

* For longer time series, LSTM and its variations were able to perform better than simpler models.

* LSTM multivariate proved to be robust when compared with other models.

* Models variations presents some advantages, however in some cases simpler models perform better.

9. References