Fundamental Of Data science

Assignment 6 – Time Series

Soroush Heidary – 96222031

Abstract :

In this assignment we're going to delve into the time series concepts and try to apply them a data set[1]

The dataset consists of these features :

- No: row number
- year: year of data in this row
- month: month of data in this row
- day: day of data in this row
- hour: hour of data in this row
- pm2.5: PM2.5 concentration
- DEWP: Dew Point
- TEMP: Temperature

[1] Benjing PM2.5 Dataset

- PRES: Pressure
- cbwd: Combined wind direction
- Iws: Cumulated wind speed
- Is: Cumulated hours of snow
- Ir: Cumulated hours of rain

We'll preprocess it & our target for forecasting would be the pollution feature (pm2.5)

This is a glimps of our dataframe after some data cleaning (not much cleaning was needed, just the datetime types had to be merged and used as index)

	pollution	dew	temp	press	wnd_spd	snow	rain
2010-01-03	60.571429	-18.255952	-10.202381	1027.910714	43.859821	2.125000	0.000000
2010-01-10	70.946429	-19.035714	-10.029762	1030.035714	45.392083	0.000000	0.000000
2010-01-17	163.660714	-12.630952	-4.946429	1030.386905	17.492976	0.000000	0.000000
2010-01-24	41.017857	-17.404762	-2.672619	1026.196429	54.854048	0.000000	0.000000
2010-01-31	53.410714	-17.565476	-2.083333	1025.273810	26.625119	0.000000	0.000000
...	...	...	...	...	...	...	...
2014-11-30	84.250000	-14.988095	-0.446429	1025.476190	76.424643	0.000000	0.005952
2014-12-07	79.255952	-14.375000	-1.303571	1029.607143	22.218869	0.005952	0.000000
2014-12-14	75.410714	-15.916667	-2.244048	1028.380952	59.875893	0.000000	0.000000
2014-12-21	48.690476	-15.089286	-0.505952	1027.339286	98.139345	0.000000	0.000000
2014-12-28	136.902778	-13.013889	-0.381944	1024.083333	34.584375	0.000000	0.000000

Theory :

Before we delve into the code, these concept deserves some explanations:

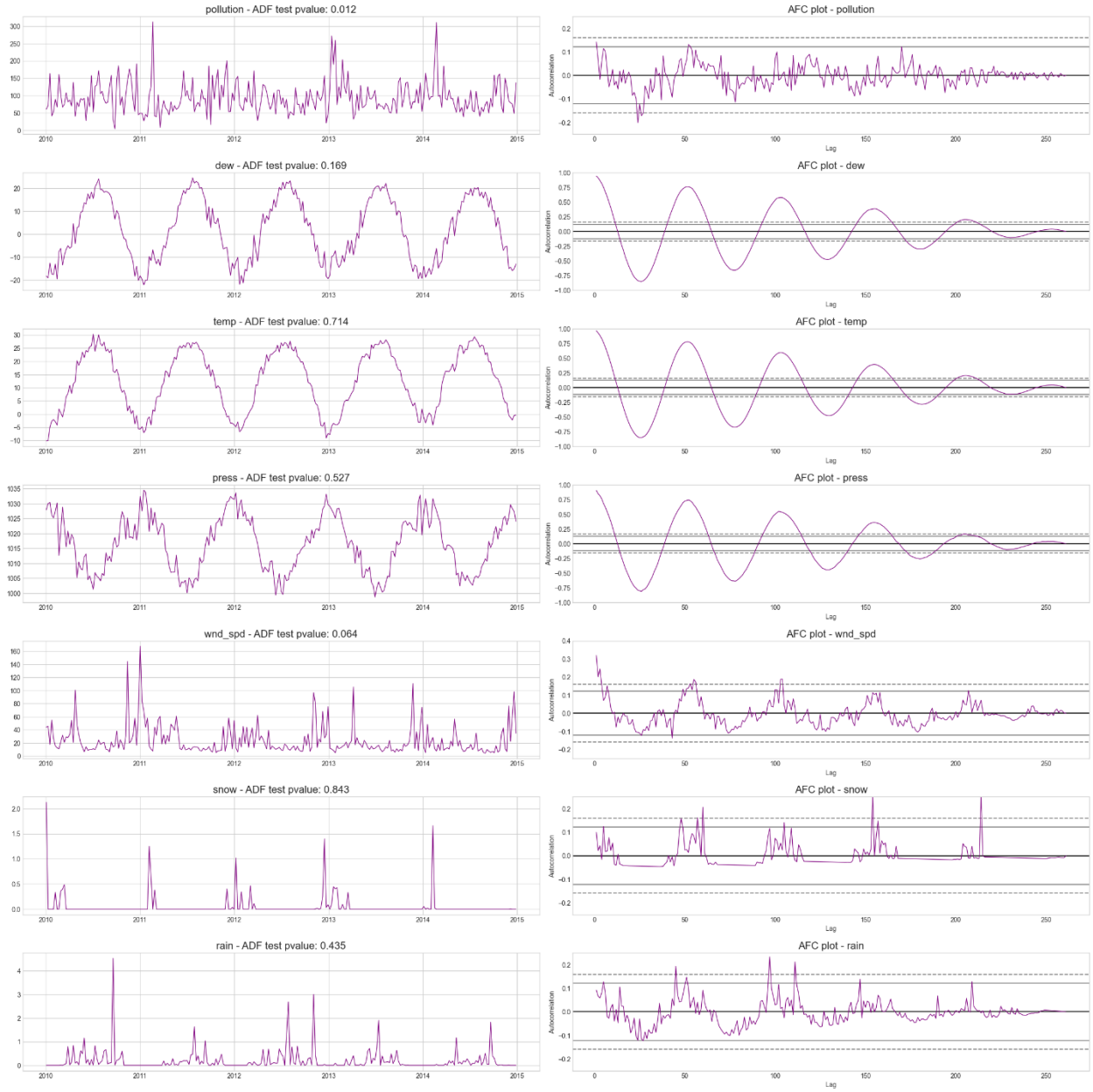
- Trend : a trend in a time series, is basically a simple linear regression which fits the time series the best, it captures if our series acts as a monotone increasing or decreasing manner.
- Seasonality : seasonality is simply a periodic behavior which a time series could follow, we can use ACF (Auto Correlation Function) also called correlogram plots to see whether or not a time series has seasonality or not
- Unit root : a unit root is hard to capture by the eye, though in theory pretty easy to test, it is merely a factor which would yield non-stationary time-series if one contains it. For example in a simple AR(1) model it is said to have a unit root if its coefficient (only coefficient) is equal to 1
- Stationarity : a time series is said to be stationary if it fulfills 3 criteria:
    - Has a constant mean over different time periods
    - Has a constant variance over different time periods
    - Does not include seasonality attribute

  This attribute is crucial if we're trying out models like AR & some of it's variants, MA, etc.

- Dickey Fuller Test : one naïve way to check for stationarity is to just visually analyze it, although there are more precise methods, like DF Test (Dickey Fuller Test) which statistically tests if our series has a unit root or not, comparing the t-test of our coefficients with a dickey fuller distribution which would be the criteria to reject if a time series is stationary or not. This model tests the stationarity in a AR(1) model, for more complicated models Augmented DF Test is introduced.

Data Insights :

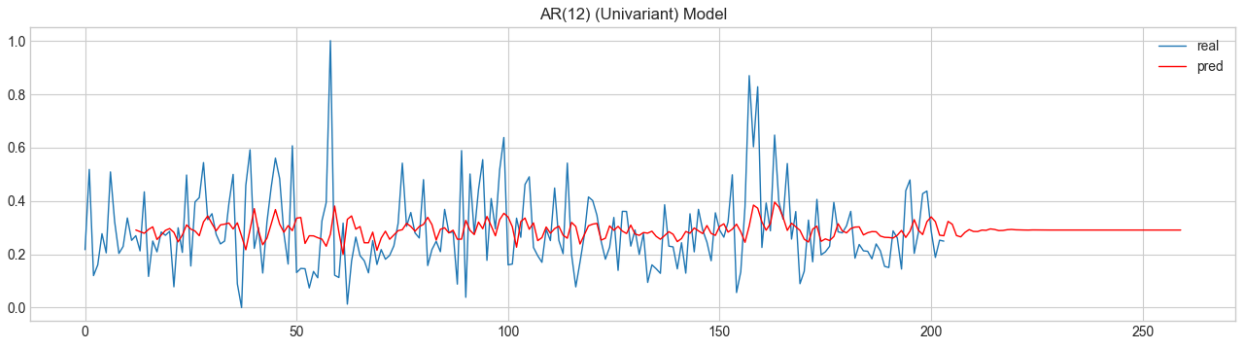
Data was recorded hourly, but for the sake of better visualizations and simplicity I have aggregated them by their mean to weekly records, below we can see each feature vs time plus their AFC plot (Autocorrelation map) the AFC is created using all the lags possible. (p-value<0.05 could be a good threshold for stationarity)



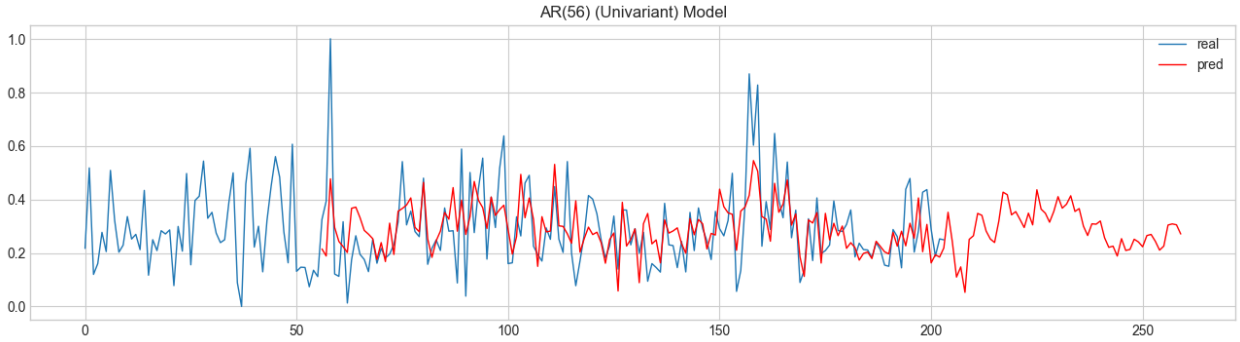
Models : the models are trained with normalized data (AR and it's variants proved to work the same with the un-normalized data, though for the sake of comparing all of the models using their final loss, all the models are train on the normalized data)

Also to visualize the results we will see each models prediction on a portion of the training set, and a year ahead (56 weeks ahead of our data set, which is produced by the model)

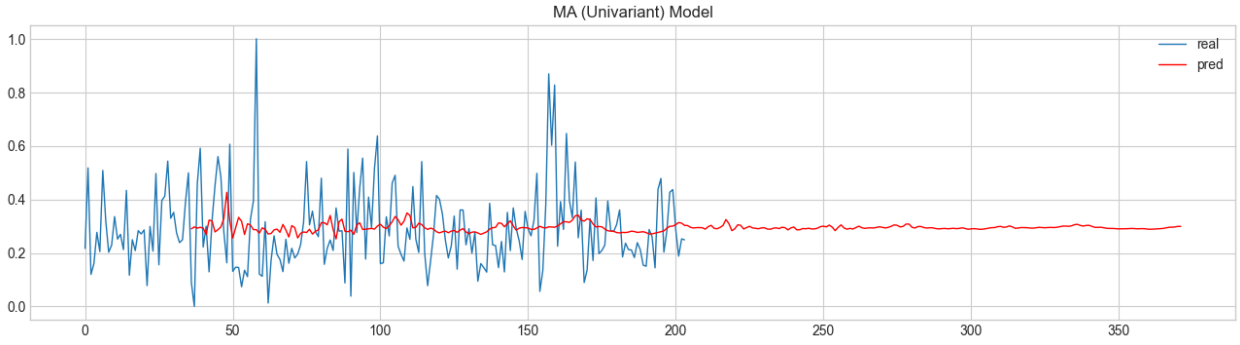
AR(Auto Regression): The method is suitable for univariate time series without trend and without seasonal components, when the series is stationary, using 12 and 56 lags on each model, the results are as below:



The AR(12) performs poorly as 12 lags(1 Season) are seemingly not enough to predict how the next week's pollution would be, though 56 lags (1 Year) is exatcly what we're looking for

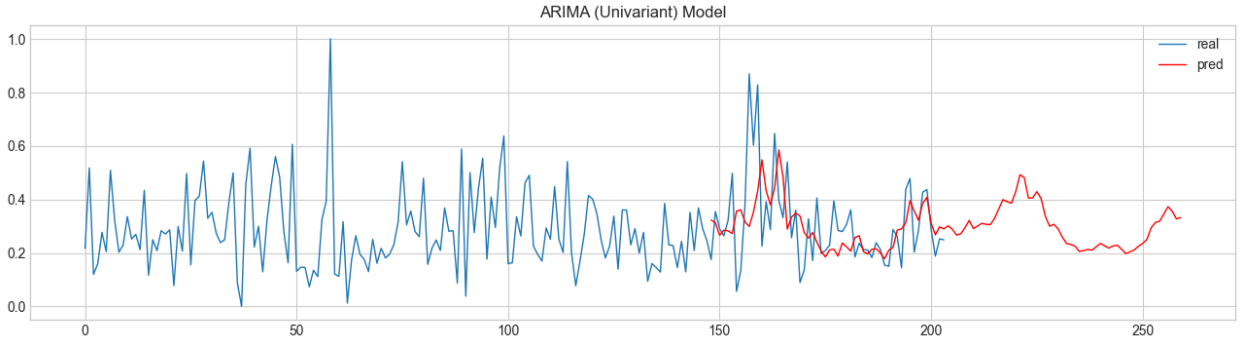


MA (Moving Average): using moving average technique alone is expected to poor results, also using MA alone is expected to yield results which fluctuate less that AR Models this could possibly yield less error as the model could just learn to forecast next weeks pollution to be around the mean value of all the previous weeks, and thus fluctuating less while giving less MSE error (although having low error doesn't necessarily means a good performance in time-series)

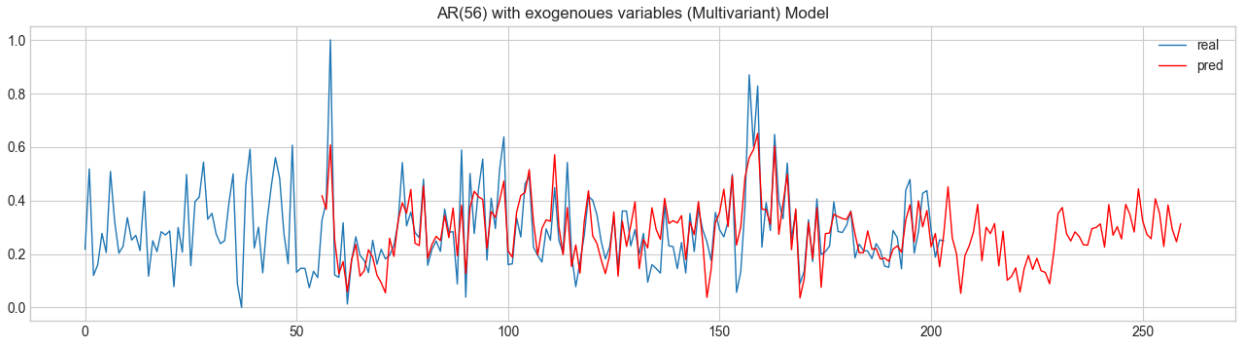


ARIMA (Auto Regression Integrated Moving Average): as expected using only MA to forecast yielded poor results, so using a combination of AR & MA could yield more suitable results, the "Integration" part is a means of differencing pre-processing step of the sequence to make the sequence stationary so AutoRegression is sufficiently good

when we're working with stationary series (which we are not, and that's why visually the AR(56) models shows us the best performance)

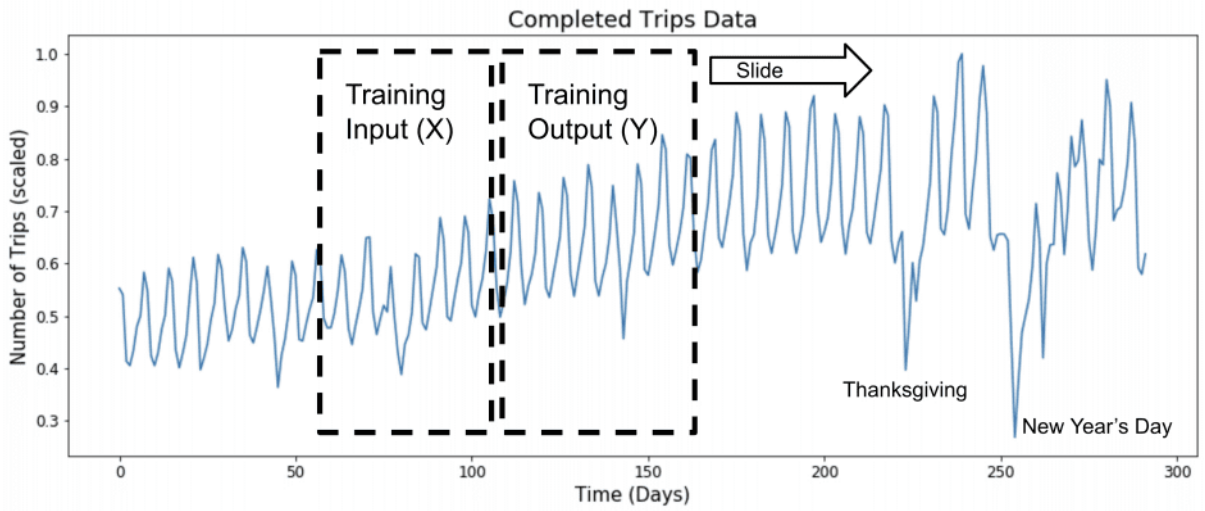


The Multivariant AR Model : we could actually use other features to affect the prediction on the pollution, feeding the other features as exogenous data to the AR model could do the trick, this helps the result to further improve, here are the results:



An improvement is clearly seen between this and the AR(56) Univariant model, we'll see the errors of each model later on

RNN: for this model(also for LSTM and CNN) we're using this technique for the test-train split of the data (image source):



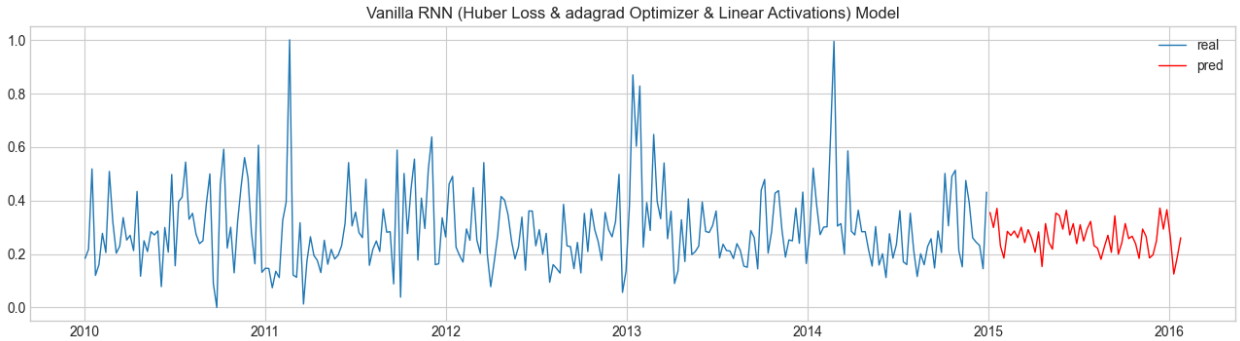
The window which slides though the series was equal to 56 weeks in our case (4 and 12 were tried out, though 56 yielded the best results by far).

All the nets we'll see below use these hyper parameters:

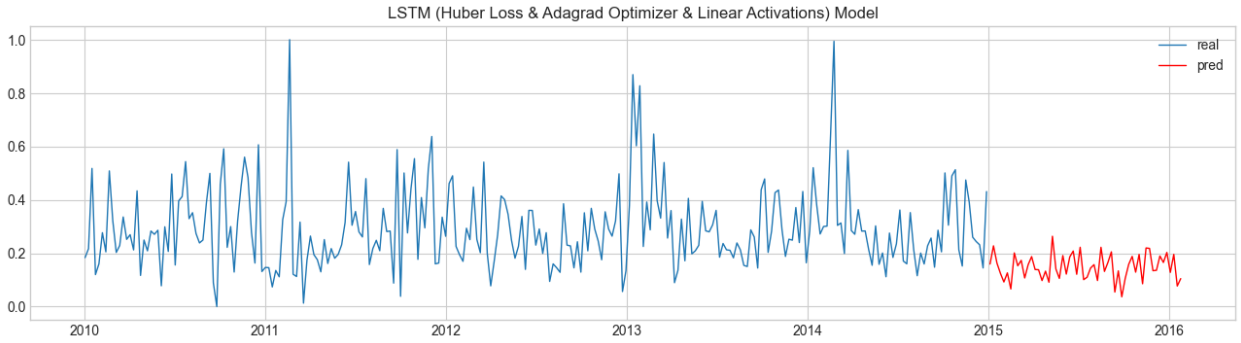
- Huber Loss
- AdaGrad Optimizer
- Linear Activations (except CNN which uses tanh Activation)

The models could be further tuned to yield better results! Not much time is put into the tuning part!

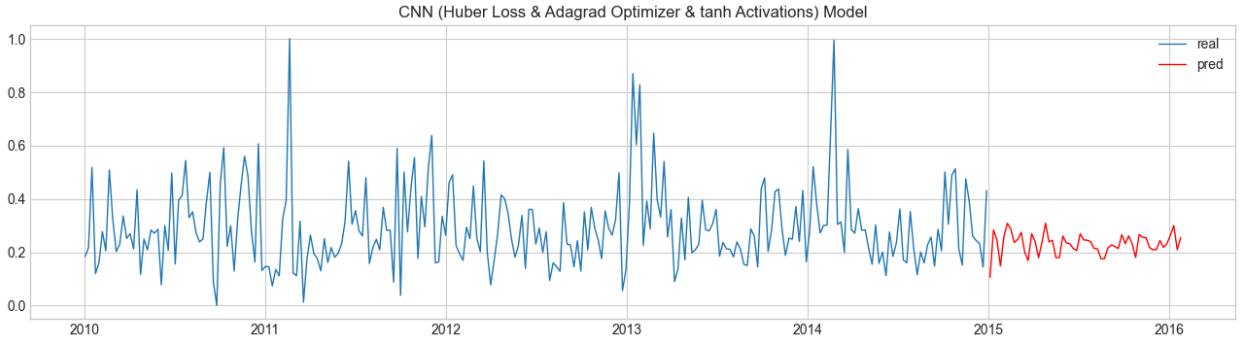
Here are the results on the Vanilla RNN :



LSTM: although LSTMs and GRUs are though of as a better version of Vanilla RNNs, here in our case this wouldn't make much of a difference, and might even yield worse results, the reason is intuitive, we don't really need much long term memory when the timeseries is much more dependent on the closer lags(like 1 month ago), than those before(8 month ago), this reasoning is just provided for this specific task (pollution forecasting!)



CNN (Convolution 1D): using a CNN is not much different than the 2 previous nets as we're dealing with univariant data.



We can see each model's performance using Huber Loss below, the reason I used hubber loss was that, in some of the models MSE & MAE loss functions would have made the

model to learn to forecast only values close to the mean of the series, which could be handled with Huber Loss a little better than others

Losses :

Model	Huber Loss
AR(56) Multivariant	0.0042
ARIMA	0.0046
AR(56) Univariant	0.0077
Vanilla RNN	0.0135
AR(12)	0.0178
LSTM	0.0239
CNN	0.0457
MA	0.0521

Conclusion :

using classic time series forecasting models like AR and it's variants did prove to outperformed Neural Nets when dealing with stationary time series, the data set I used was stationary from the beginning with out any data engineering, so using a simple AR model would be suffiently good to forecast the series (using a multivariant version outperformed all the other models), ARIMA which combines both MA and RA outperformed univariant AR, and surprisingly the RNN performed better than the LSTM,

although the neural nets I trained here could be further more tuned to even outperform the AR variants, not much time was put into the neural nets, this is one of the reasons they were outperformed by the AR variants (a recent publication by the UBER research team proves this assumption: link), also the Prophet library was tried out too, which almost outperformed every model I have shown here, though I couldn't install the library on my computer and don't have any results saved to put in the report :P

Recourses :

A combination of all the links below where used for this assignment:

https://machinelearningmastery.com/time-series-forecasting-methods-in-python-cheat-sheet/

https://machinelearningmastery.com/lstm-model-architecture-for-rare-event-time-series-forecasting/

https://machinelearningmastery.com/convert-time-series-supervised-learning-problem-python/

https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/

https://towardsdatascience.com/7-tips-to-choose-the-best-optimizer-47bb9c1219e

https://www.kaggle.com/ryanholbrook/forecasting-with-machine-learning