Air Pollution Time Series

Time series prediction using air pollution dataset from the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/PM2.5+Data+of+Five+Chinese+Cities) that measures hourly air quality in Chinese cities/city districts.

	No	year	month	day	hour	season	PM_Jingan	PM_US Post	PM_Xuhui	DEWP	HUMI	PRES	TEMP	cbwd	Iws	precipitation	Iprec
0	1	2010	1	1	0	4	NaN	NaN	NaN	-6.0	59.48	1026.1	1.0	cv	1.0	0.0	0.0
1	2	2010	1	1	1	4	NaN	NaN	NaN	-6.0	59.48	1025.1	1.0	SE	2.0	0.0	0.0
2	3	2010	1	1	2	4	NaN	NaN	NaN	-7.0	59.21	1025.1	0.0	SE	4.0	0.0	0.0
3	4	2010	1	1	3	4	NaN	NaN	NaN	-6.0	63.94	1024.0	0.0	SE	5.0	0.0	0.0
4	5	2010	1	1	4	4	NaN	NaN	NaN	-6.0	63.94	1023.0	0.0	SE	8.0	0.0	0.0

The from Shanghai city was used. The dataset contains 17 features, 3 of which are PM 2.5 concentration measurements. The time period is between Jan 1st, 2010 to Dec 31st, 2015.

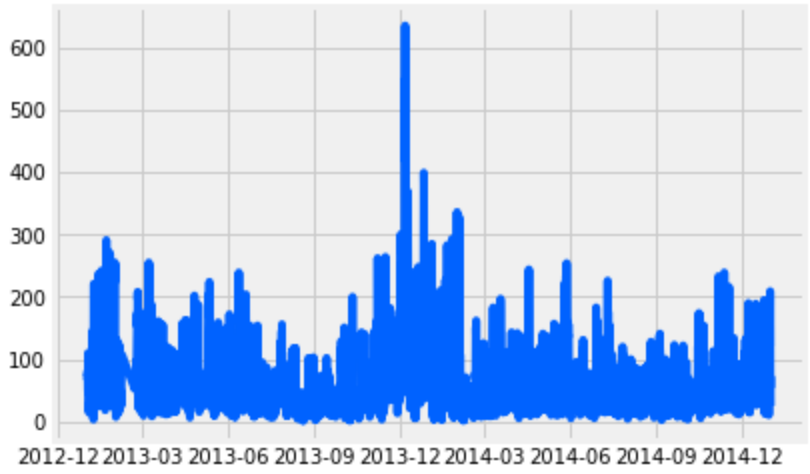
1. No: row number
2. year: year of data in this row
3. month: month of data in this row
4. day: day of data in this row
5. hour: hour of data in this row
6. season: season of data in this row
7. PM: PM2.5 concentration (ug/m^3)
8. DEWP: Dew Point (Celsius Degree)
9. TEMP: Temperature (Celsius Degree)
10. HUMI: Humidity (%)
11. PRES: Pressure (hPa)
12. cbwd: Combined wind direction
13. Iws: Cumulated wind speed (m/s)
14. precipitation: hourly precipitation (mm)
15. Iprec: Cumulated precipitation (mm)

In this project, the PM 2.5 measurement from Jingan is used.

Exploratory Data Analysis

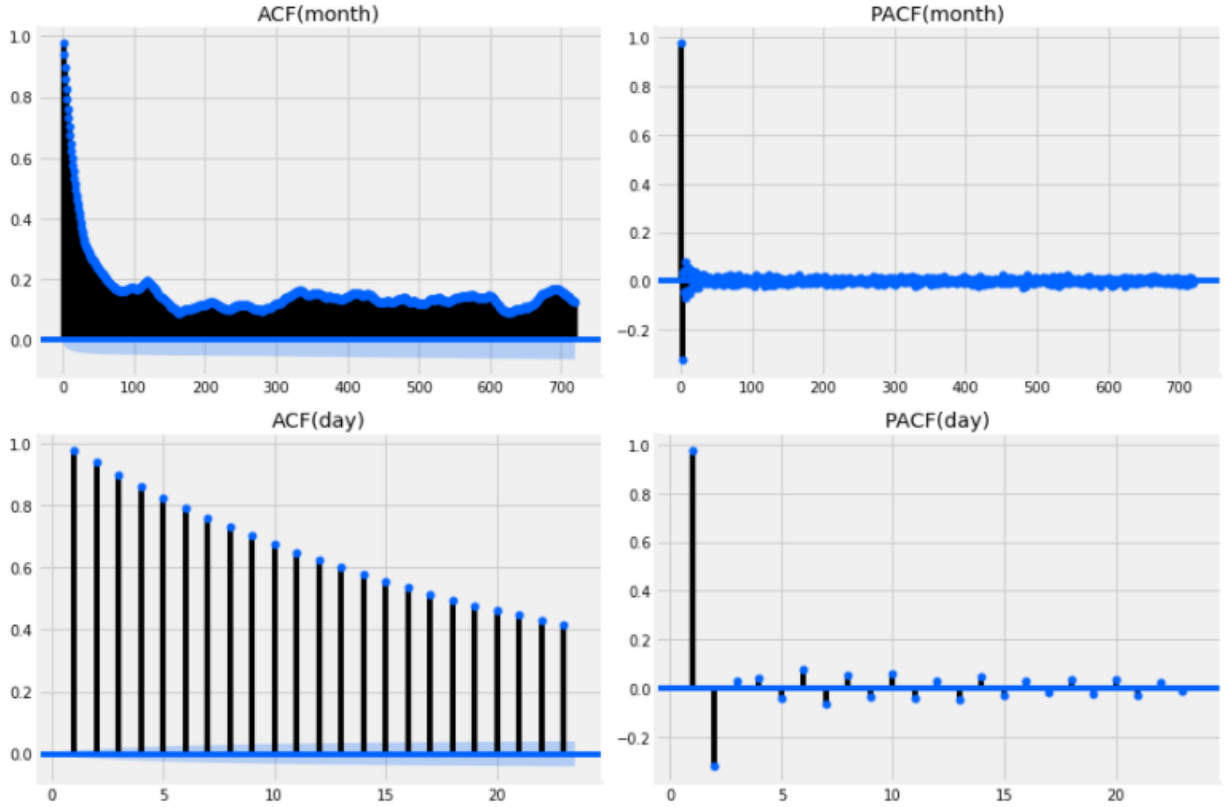
The steps taken for data analysis were,

- Interpolating null values
- Creating a DateTime column using the existing columns of date and time
- Creating a dataset using only hourly PM values and Datetime values as index
- Train-Test split
    - Using values from 1st Jan 2013 to 30th Nov 2014 as training data
    - Using Dec 2014 as test data



A visual representation of the final 2-year data is shown. There seems to be seasonality with higher spikes during winter (Jan and Dec). Some values in the middle in Dec 2013 seem to be outliers.

Augmented Dickey-Fuller Test (ADF Test) results gave am ADF statistic value of -12.67 and a p-value of 1.19e-23. Since this p-value is less than 0.05, it suggests that the null hypothesis (Ho: Series is non-stationary) is rejected. Although this test cannot be the sole criteria to decide whether the series is stationary. The ACF and PACF plots are investigated for 24 lags (1 day) and 30*12 lags (1 month)



Box-Jenkins Method

ACF Shape	Indicated Model
Exponential, decaying to zero	Autoregressive model. Use the partial autocorrelation plot to identify the order of the autoregressive model.
Alternating positive and negative, decaying to zero	Autoregressive model. Use the partial autocorrelation plot to help identify the order.
One or more spikes, rest are essentially zero	Moving average model, order identified by where plot becomes zero.
Decay, starting after a few lags	Mixed autoregressive and moving average (ARMA) model.
All zero or close to zero	Data are essentially random.
High values at fixed intervals	Include seasonal autoregressive term.
No decay to zero	Series is not stationary.

There is an exponential decay in the ACF plot (not exactly to zero though, to <0.2) with lag 1 and 2 in PACF having significantly large values (>0.1). There is also an alternating pattern in the PACF. This suggests that the model has an AR(2) component. The model could also be ARIMA because the series is non-stationary.

Models

3 types of models are explored for this dataset.

1. ARMIA
2. RNN
3. LSTM

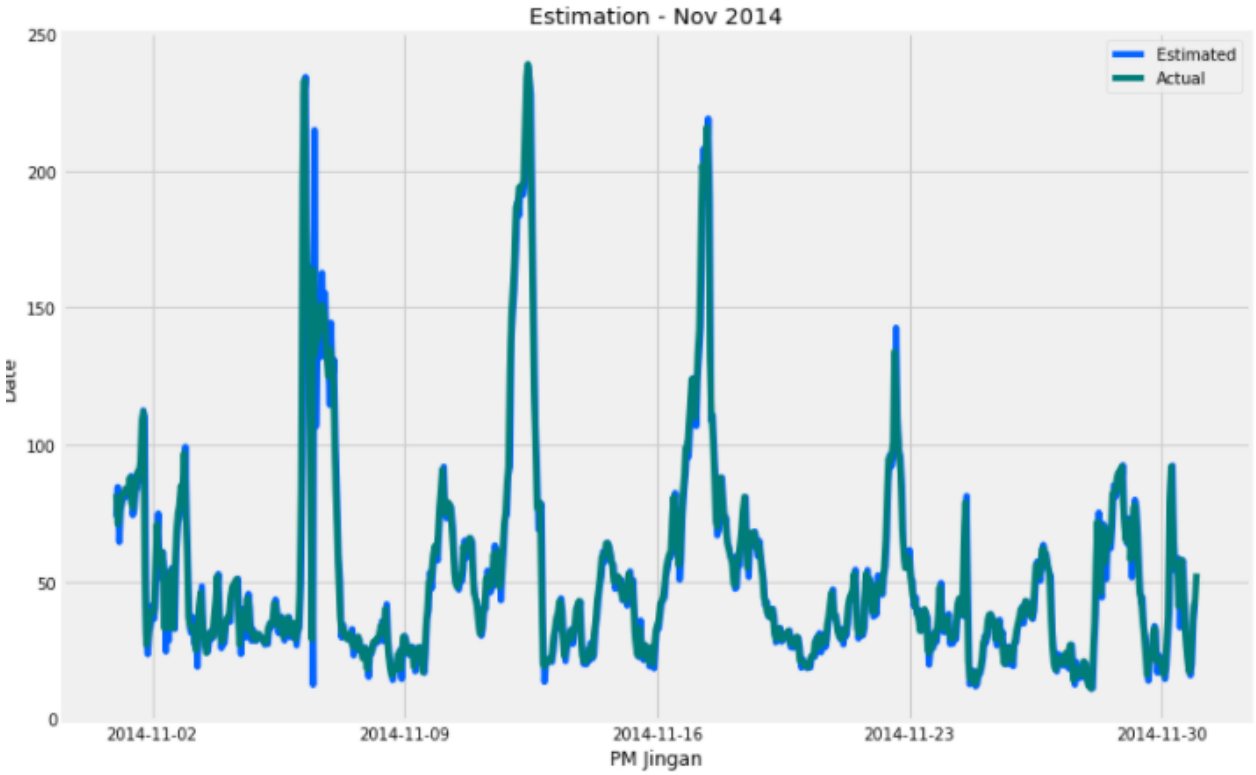
ARIMA

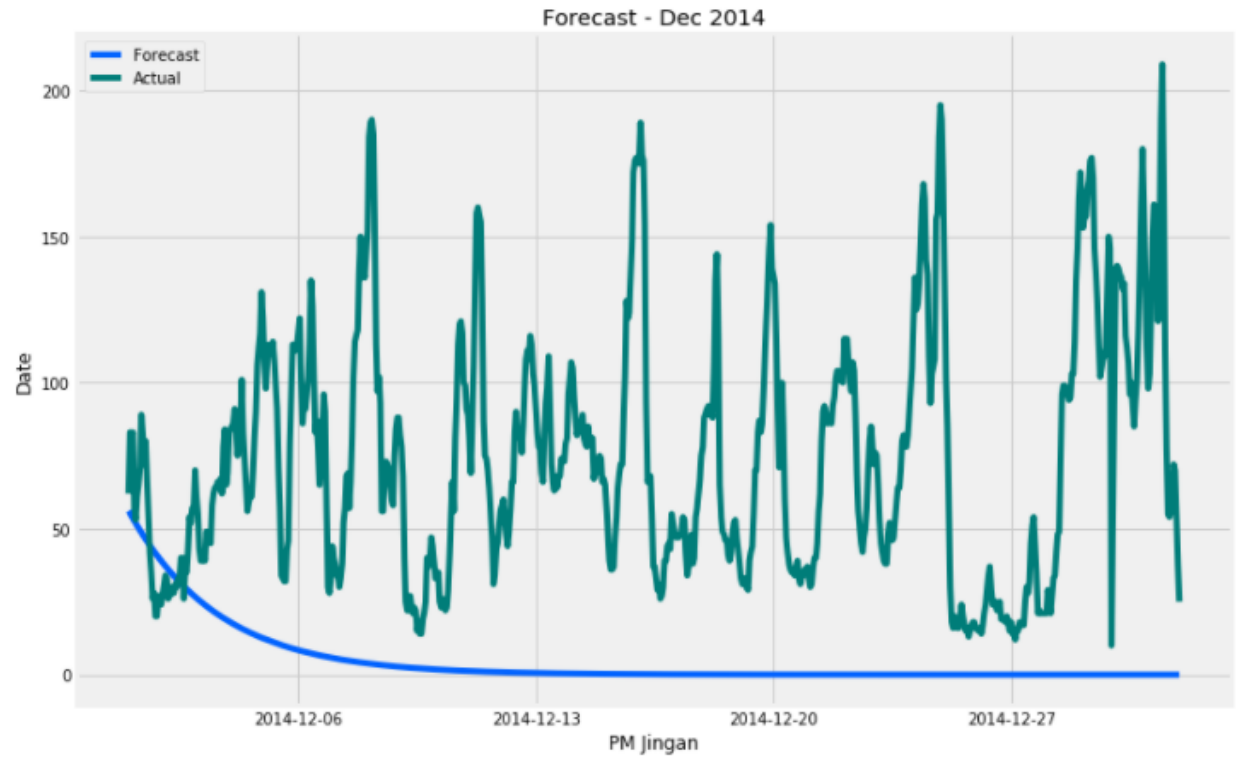
Different ARIMA(p,d,q) models were explored and their AIC recorded.

```
Fit ARIMA: order=(0, 0, 0); AIC=178790.629, BIC=178806.084, Fit time=0.539 seconds
Fit ARIMA: order=(1, 0, 0); AIC=126824.556, BIC=126847.739, Fit time=0.929 seconds
Fit ARIMA: order=(0, 0, 1); AIC=159950.704, BIC=159973.887, Fit time=1.433 seconds
Fit ARIMA: order=(2, 0, 0); AIC=125054.895, BIC=125085.806, Fit time=0.354 seconds
Fit ARIMA: order=(2, 0, 1); AIC=125047.472, BIC=125086.111, Fit time=5.054 seconds
Fit ARIMA: order=(1, 0, 1); AIC=125196.309, BIC=125227.219, Fit time=2.366 seconds
Fit ARIMA: order=(2, 0, 2); AIC=124453.696, BIC=124500.063, Fit time=6.809 seconds
Fit ARIMA: order=(1, 0, 2); AIC=125006.070, BIC=125044.709, Fit time=2.327 seconds
Total fit time: 20.027 seconds
124453.69629207299

Fit ARIMA: order=(0, 1, 0); AIC=127005.483, BIC=127020.938, Fit time=0.110 seconds
Fit ARIMA: order=(1, 1, 0); AIC=125408.155, BIC=125431.338, Fit time=0.156 seconds
Fit ARIMA: order=(0, 1, 1); AIC=125474.169, BIC=125497.352, Fit time=0.111 seconds
Fit ARIMA: order=(2, 1, 0); AIC=125371.705, BIC=125402.615, Fit time=0.162 seconds
Fit ARIMA: order=(2, 1, 1); AIC=125013.653, BIC=125052.291, Fit time=5.451 seconds
Fit ARIMA: order=(1, 1, 1); AIC=125383.509, BIC=125414.420, Fit time=2.502 seconds
Fit ARIMA: order=(2, 1, 2); AIC=124750.432, BIC=124796.798, Fit time=6.361 seconds
Fit ARIMA: order=(1, 1, 2); AIC=124752.605, BIC=124791.243, Fit time=5.733 seconds
Total fit time: 20.871 seconds
124750.43245588645
```

ARIMA(2,0,2) was found to have the lowest AIC and hence this model was chosen. This is essentially the same as ARMA(2,2). A model was trained and tested.
The model performed well for estimating seen data (Nov 2014) with an MSE of 129.97.
However, for forecasting unseen testing data (Dec 2014), the model gave an MSE of 6858.14.





RNN

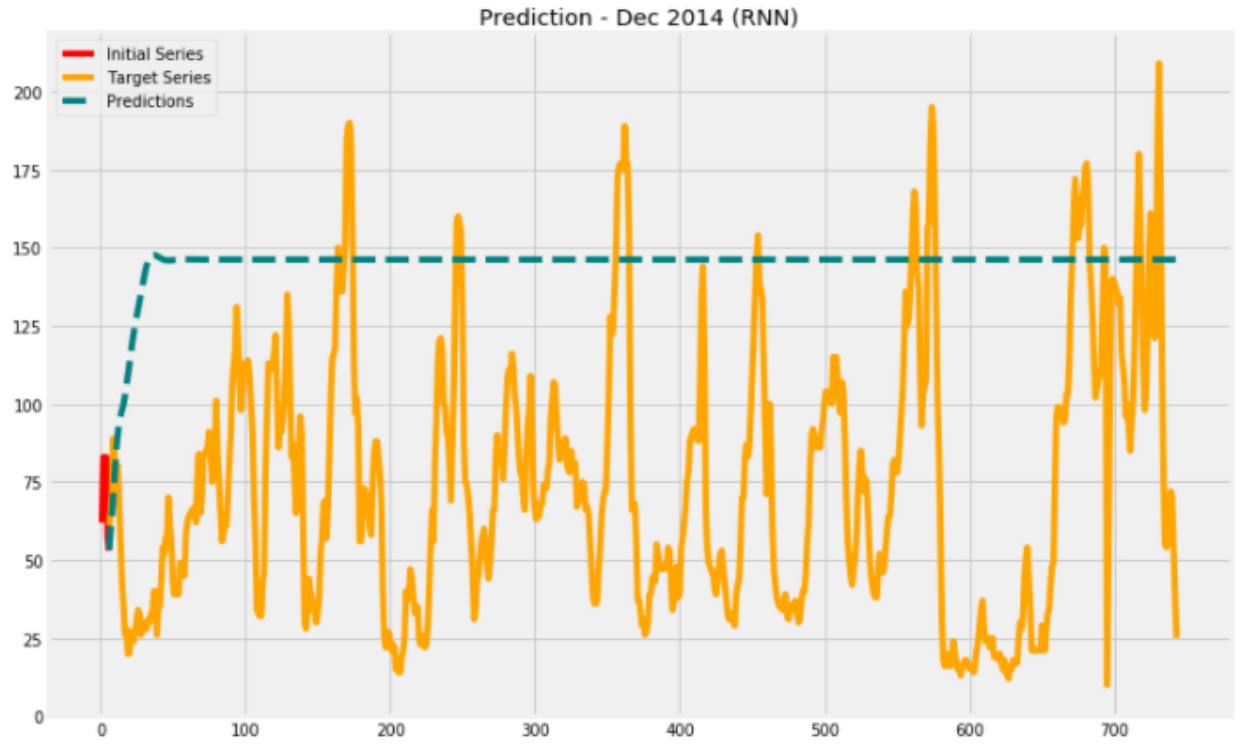
Before using the data for neural network techniques, the data were scaled using MinMax.

Training and testing datasets were prepared using a sample gap of 3 and input hours as 6. This means that 6 hours of data was used to predict the 7th hour, and the pattern was repeated after a gap of every 3 hours.

Layer (type)	Output Shape	Param #
simple_rnn_38 (SimpleRNN)	(None, 6, 90)	8280
simple_rnn_39 (SimpleRNN)	(None, 6, 120)	25320
simple_rnn_40 (SimpleRNN)	(None, 30)	4530
dense_19 (Dense)	(None, 12)	372
dense_20 (Dense)	(None, 1)	13

Total params: 38,515
Trainable params: 38,515
Non-trainable params: 0

The RNN model was as shown. The model was fit for 100 epochs using a batch size of 50, with *'adam'* optimizer and *'mean squared error'* loss. For predicting test data values, the model gave an MSE of 6770.02.

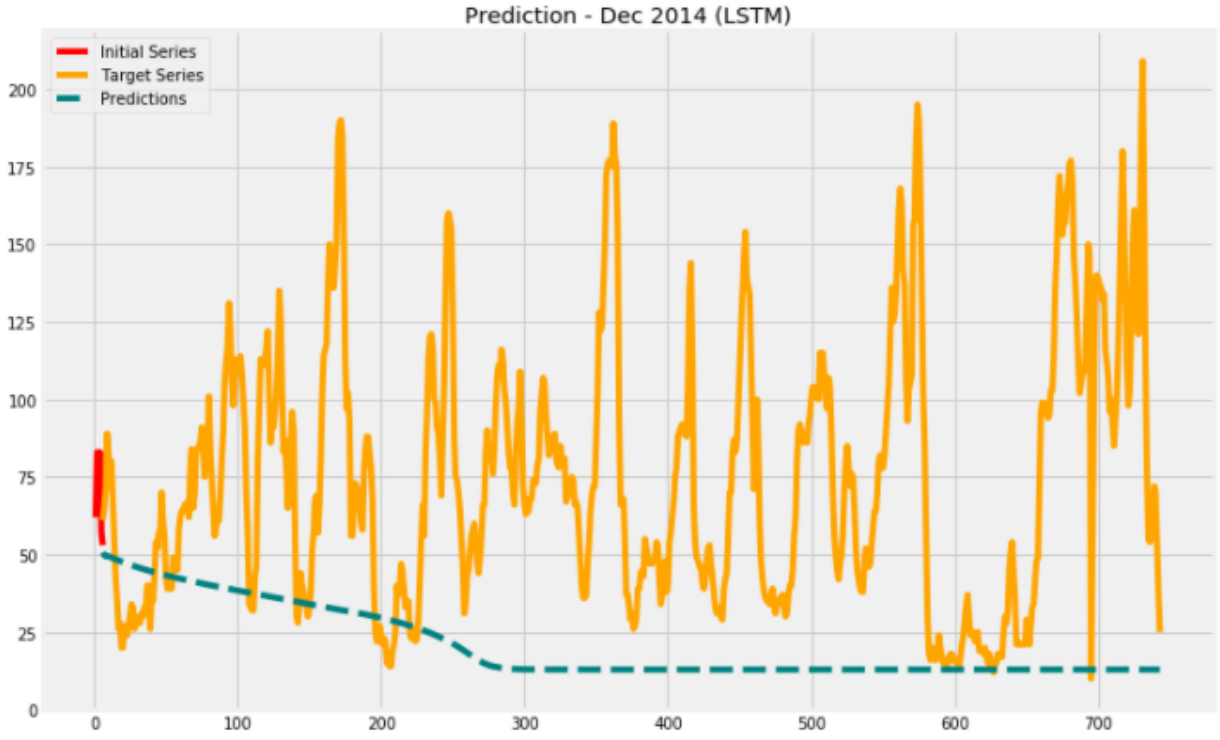


LSTM

The same parameters and model size was used for LSTM. The model gave an MSE of 4900 on the test data, which is clearly an improvement as compared to RNN.

Layer (type)	Output Shape	Param #
lstm_7 (LSTM)	(None, 6, 90)	33120
lstm_8 (LSTM)	(None, 6, 120)	101280
lstm_9 (LSTM)	(None, 30)	18120
dense_27 (Dense)	(None, 12)	372
dense_28 (Dense)	(None, 1)	13

Total params: 152,905
Trainable params: 152,905
Non-trainable params: 0



Conclusion

Although none of the models gave satisfactory results, LSTM performed the best on the basis of MSE. The LSTM model also performed much better than the RNN, given the same parameters and model size, although LSTM had many more trainable parameters. All three models gave an acceptable prediction only for the first few prediction values, after which they saturated to a fixed value.

In order to improve the performance, hyperparameters of the LSTM and RNN models could be tuned and different model sizes could be tried.