HW6

FORECAST DAILY AVERAGE PRESSURE (PRES) IN TIANTAN, BEIJING IN 2017 MARCH

Dhanabordee Mekintharanggur
6238077121

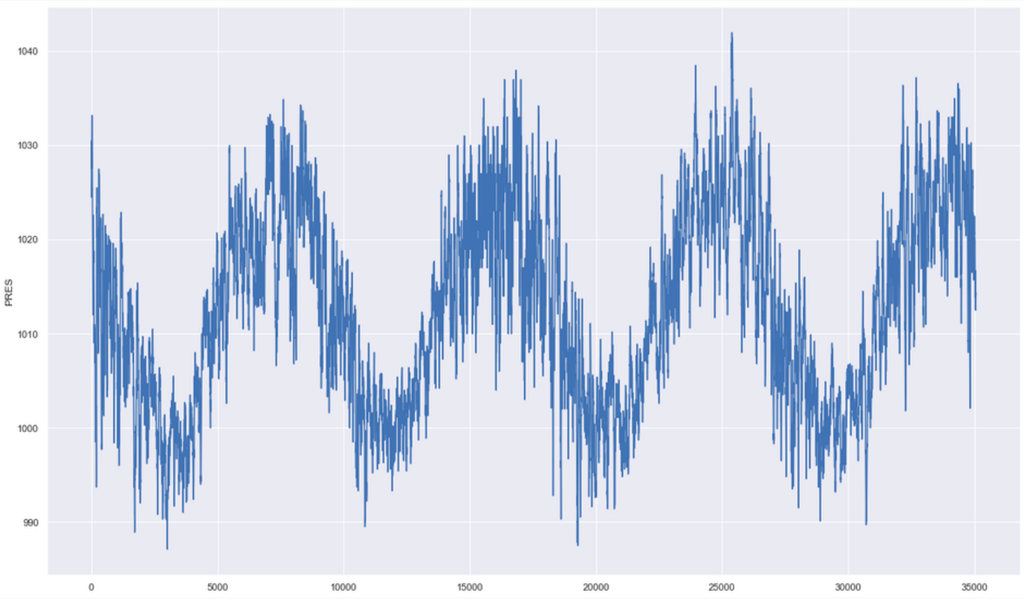
Step 1 - Loading the Data

The data is downloaded as a .csv file format and is placed in the same directory as the .ipynb file

The csv is read using pd.read_csv to read the data in the dataframe format for further exploration and preprocessing

	No	year	month	day	hour	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	RAIN	wd	WSPM	station
0	1	2013	3	1	0	6.0	6.0	4.0	8.0	300.0	81.0	-0.5	1024.5	-21.4	0.0	NNW	5.7	Tiantan
1	2	2013	3	1	1	6.0	29.0	5.0	9.0	300.0	80.0	-0.7	1025.1	-22.1	0.0	NW	3.9	Tiantan
2	3	2013	3	1	2	6.0	6.0	4.0	12.0	300.0	75.0	-1.2	1025.3	-24.6	0.0	NNW	5.3	Tiantan
3	4	2013	3	1	3	6.0	6.0	4.0	12.0	300.0	74.0	-1.4	1026.2	-25.5	0.0	N	4.9	Tiantan
4	5	2013	3	1	4	5.0	5.0	7.0	15.0	400.0	70.0	-1.9	1027.1	-24.5	0.0	NNW	3.2	Tiantan

Notice that there are NaN values present in multiple columns. This has to be dealt with in the next step.



A quick lineplot shows the hourly pressure measured for a total of approximately 35,000 data points that has a cyclical pattern that repeats approximately 4 times

This points out that the data has seasonality and a potentially slight upward trend as the cycles get slightly higher highs. Further investigation is needed later.

Step 2 - Exploring and Cleaning the Data

There are a total of 1461 rows of data

Multiple columns contain null values

The 'PRES' column has a total of 20 data points missing

```
Data columns (total 18 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   No       35064 non-null  int64
 1   year     35064 non-null  int64
 2   month    35064 non-null  int64
 3   day      35064 non-null  int64
 4   hour     35064 non-null  int64
 5   PM2.5    34387 non-null  float64
 6   PM10     34467 non-null  float64
 7   SO2      33946 non-null  float64
 8   NO2      34320 non-null  float64
 9   CO       33938 non-null  float64
 10  O3       34221 non-null  float64
 11  TEMP     35044 non-null  float64
 12  PRES     35044 non-null  float64
 13  DEWP     35044 non-null  float64
 14  RAIN     35044 non-null  float64
 15  wd       34986 non-null  object
 16  WSPM     35050 non-null  float64
 17  station  35064 non-null  object
```

```
No            0
year          0
month         0
day           0
hour          0
PM2.5       677
PM10        597
SO2        1118
NO2         744
CO         1126
O3          843
TEMP         20
PRES         20
DEWP         20
RAIN         20
wd           78
WSPM         14
station       0
dtype: int64
```

Step 2 - Exploring and Cleaning the Data

Dealing with the missing values

- Only the PRES value will be used to create the time series. Since there is no causal relationship for PRES and its own lag values are independent variables, the time series is considered to be a univariate time series.
- The PRES column does contain 20 missing values.
- Since this is a time-series problem, the missing values can be interpolated using Panda's **.interpolate()**

```
No          0
year        0
month       0
day         0
hour        0
PM2.5     677
PM10      597
SO2      1118
NO2       744
CO       1126
O3        843
TEMP       20
PRES        0
DEWP       20
RAIN       20
wd         78
WSPM       14
station     0
dtype: int64
```

There are now no missing PRES values

Set time feature as index

- Note that the time measurement of the data is separated into 4 different columns: year, month, day, and hour.
- Begin by creating a new column whose value is a datetime derived from the year, month, day, and hour.

Next, the datetime features is set as the index.

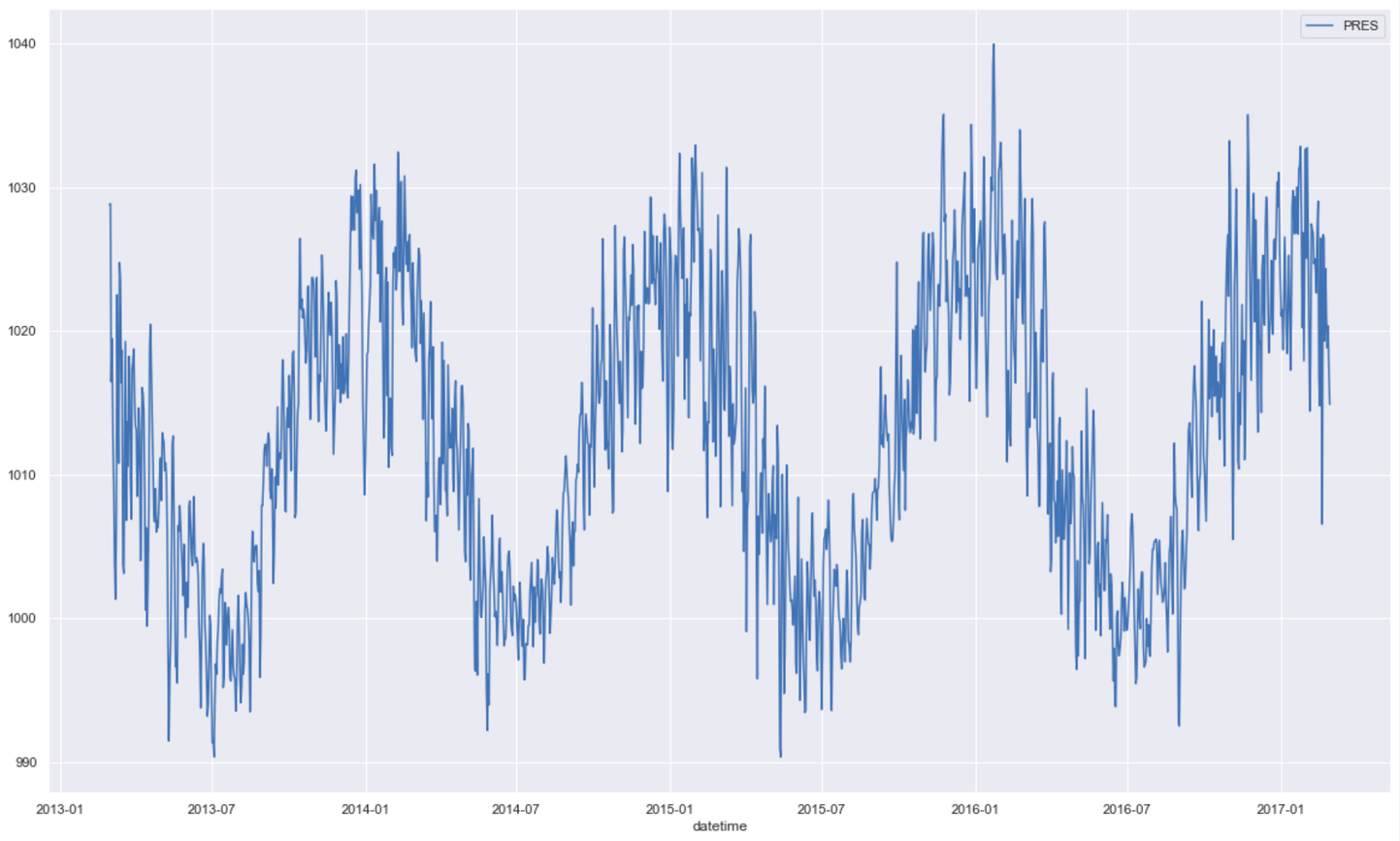
Then, the pressure is aggregated daily using the mean value of PRSE measured in each of the 24 hours in the day.

	PRES
datetime	
2013-03-01	1028.78
2013-03-02	1028.85
2013-03-03	1016.46
2013-03-04	1019.45
2013-03-05	1012.70
...	...
2017-02-24	1021.01
2017-02-25	1018.81
2017-02-26	1020.34
2017-02-27	1017.14
2017-02-28	1014.89

1461 rows × 1 columns

There are now a total of 1461 rows of data

Step 2 - Exploring and Cleaning the Data



Now, the lineplot of the daily-aggregated pressure is presented as a time series with the datetime being the x-axis and the pressure values being the y-axis.

Step 3 - Preprocess the Data

Split the test and train data

```
df_train, df_test = train_test_split(df,test_size=0.1,shuffle=False)
```

✓ 0.0s

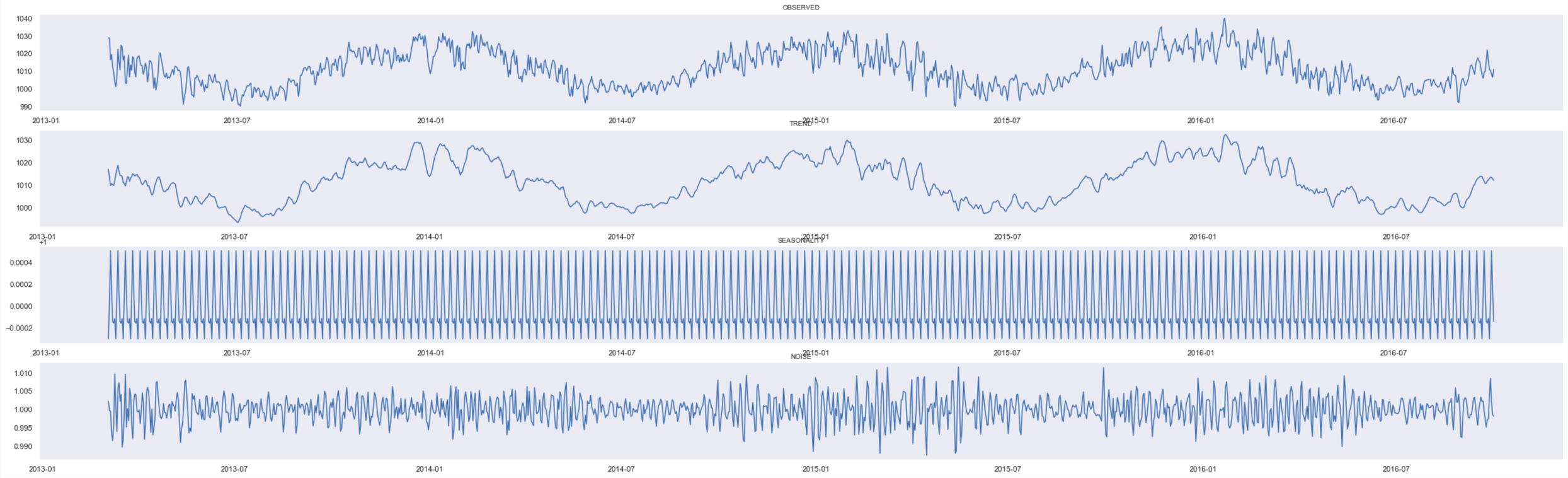
The first 1314 rows are separated into the train data, and the remaining 147 rows are used as the test data

datetime	PRES	p_lag1	p_forward1	p_ma2
2013-03-01	1028.78	NaN	1028.85	NaN
2013-03-02	1028.85	1028.78	1016.46	1028.815
2013-03-03	1016.46	1028.85	1019.45	1022.655
2013-03-04	1019.45	1016.46	1012.70	1017.955
2013-03-05	1012.70	1019.45	1009.30	1016.075
...	...	...	...	...
2016-09-30	1011.28	1015.47	1010.48	1013.375
2016-10-01	1010.48	1011.28	1008.65	1010.880
2016-10-02	1008.65	1010.48	1006.76	1009.565
2016-10-03	1006.76	1008.65	1011.16	1007.705
2016-10-04	1011.16	1006.76	NaN	1008.960

1314 rows × 4 columns

Step 3 - Preprocess the Data

Examine the seasonal decomposed time series

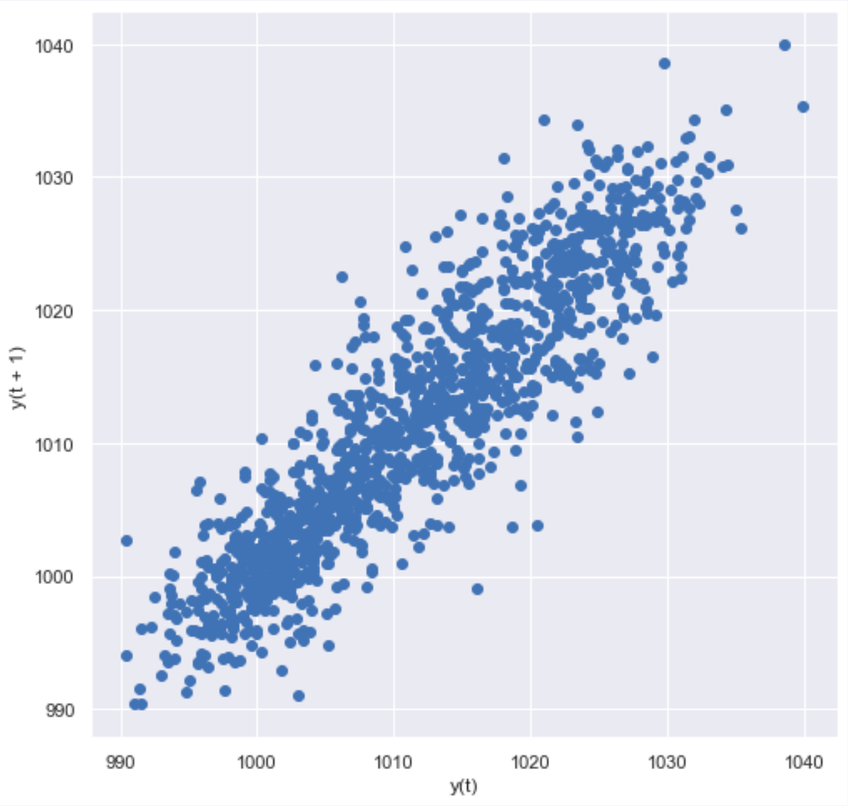


Upon closer inspection, the seasonality plot reveals that each season last 7 days rather than 365 days as originally observed from the time series plot.

This is because the trend shifts up and down throughout the entire time series.

Step 3 - Preprocess the Data

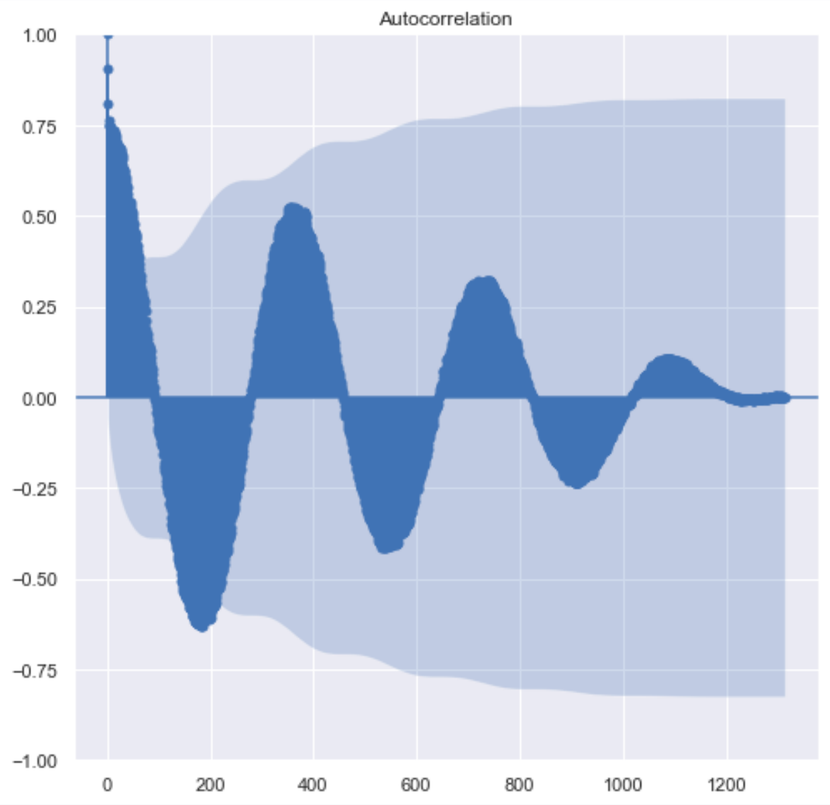
Examine the lag plot

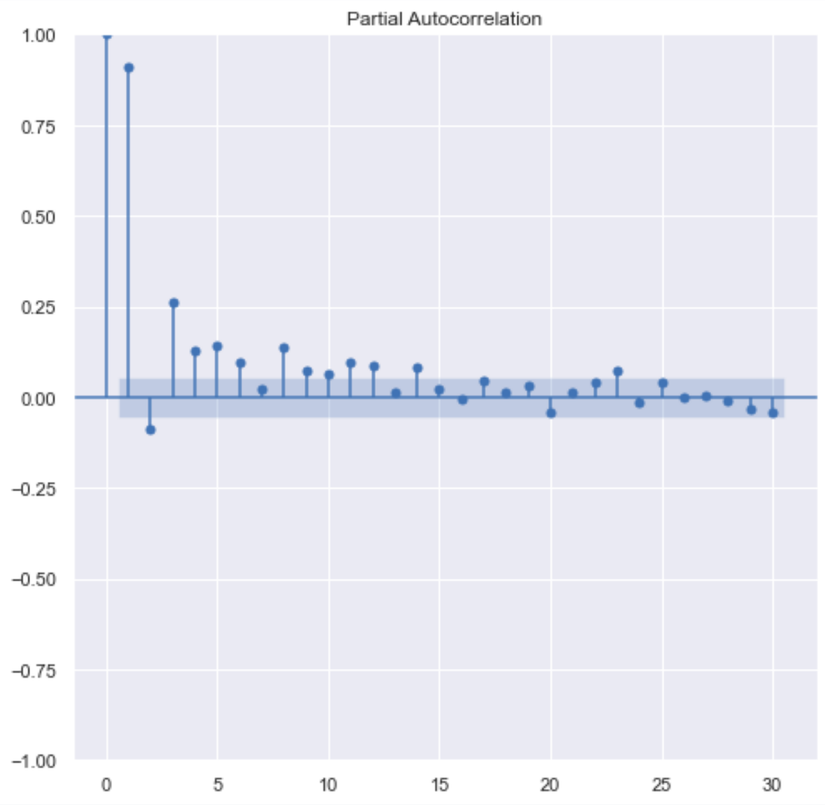


The lag plot is used to assess the presence of autocorrelation in a time series. A linear shape to the plot suggests that an autoregressive model is probably a good choice.

Step 4 - Build the AutoRegression Model

...





The ACF plot shows a tail-off behavior as no cut-offs can be observed. This shows that the time series has an AR behavior

Judging from the PACF plot, although there are multiple values that surpasses the significant threshold, there is one very clear cut-off after lag = 1

Step 4 - Build the AutoRegression Model

...

```
from statsmodels.tsa.ar_model import AutoReg
model=AutoReg(df_train['PRES'],lags=1,trend='t',seasonal=True,period=7,exog=None,missing='drop')
result=model.fit()
result.summary()
```

Using the insights from the ACF and PACF plot, the AutoRegression (AR) model is fitted with lag = 1.

From the decomposed plot of the time series earlier, the short-term season was observed where each period lasts 7 days.

AutoReg Model Results

Dep. Variable:	PRES	No. Observations:	1314
Model:	Seas. AutoReg(1)	Log Likelihood	-3719.905
Method:	Conditional MLE	S.D. of innovations	4.113
Date:	Wed, 15 Feb 2023	AIC	7459.810
Time:	13:20:58	BIC	7511.610
Sample:	03-02-2013	HQIC	7479.236
	- 10-04-2016		

The train_df contains the sample from March 2 of 2013 until April 10 of 2016. The result of fitting the AR model is as shown.

Akaike Information Criterion (AIC) = 7459.810
Bayesian Information Criterion (BIC) = 7479.236

Step 4 - Build the AutoRegression Model

...

```
from statsmodels.tsa.ar_model import AutoReg
model=AutoReg(df_train['PRES'],lags=1,trend='t',seasonal=True,period=365,exog=None,missing='drop')
result=model.fit()
result.summary()
```

Using the insights from the ACF and PACF plot, the AutoRegression (AR) model is fitted with lag = 1.

From the plot of the time series earlier, the long-term yearly cycle is the most apparent, thus I have decided to use period = 365 for our daily pressure data to fit the model with the train dataframe.

AutoReg Model Results

Dep. Variable:	PRES	No. Observations:	1314
Model:	Seas. AutoReg(1)	Log Likelihood	-3413.119
Method:	Conditional MLE	S.D. of innovations	3.256
Date:	Wed, 15 Feb 2023	AIC	7562.238
Time:	13:12:29	BIC	9468.504
Sample:	03-02-2013	HQIC	8277.122
	- 10-04-2016		

The train_df contains the sample from March 2 of 2013 until April 10 of 2016. The result of fitting the AR model is as shown.

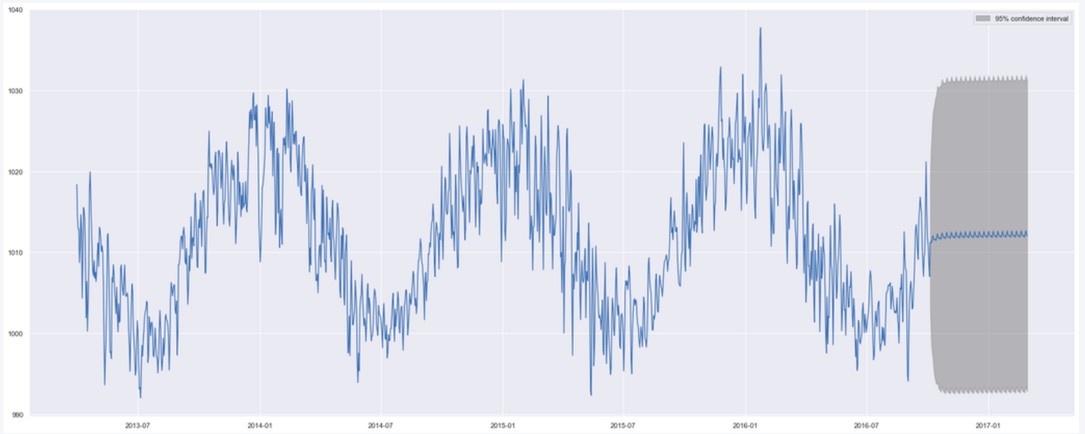
Akaike Information Criterion (AIC) = 7562.238
Bayesian Information Criterion (BIC) = 8277.122

Due to the increased lag from 7 to 365, the AIC remains very similar and increases very marginally.
However, due to the increased model complexity from the larger lag, BIC increases quite significantly to 8277.112.

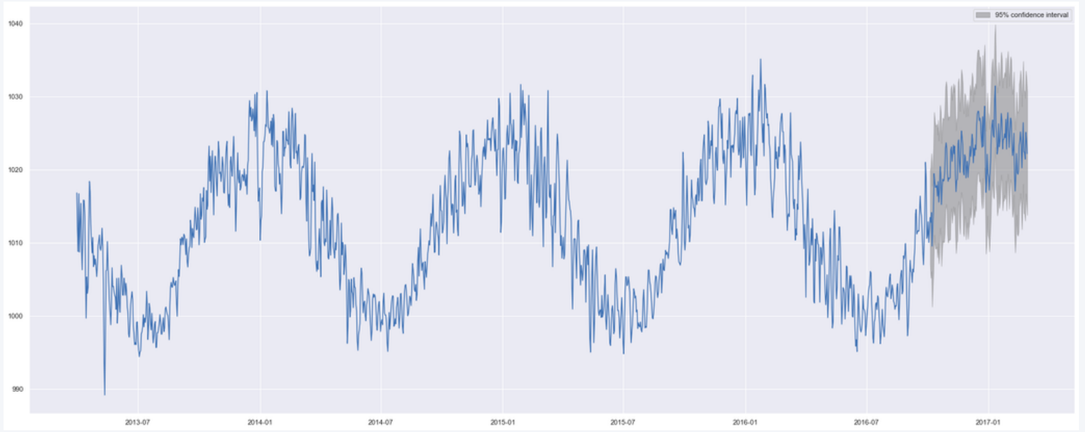
This may suggest that AR(1) with lag = 7 is the better model. However, I should not discard the possibility of lag = 365 yet ...

Step 4 - Build the AutoRegression Model

...



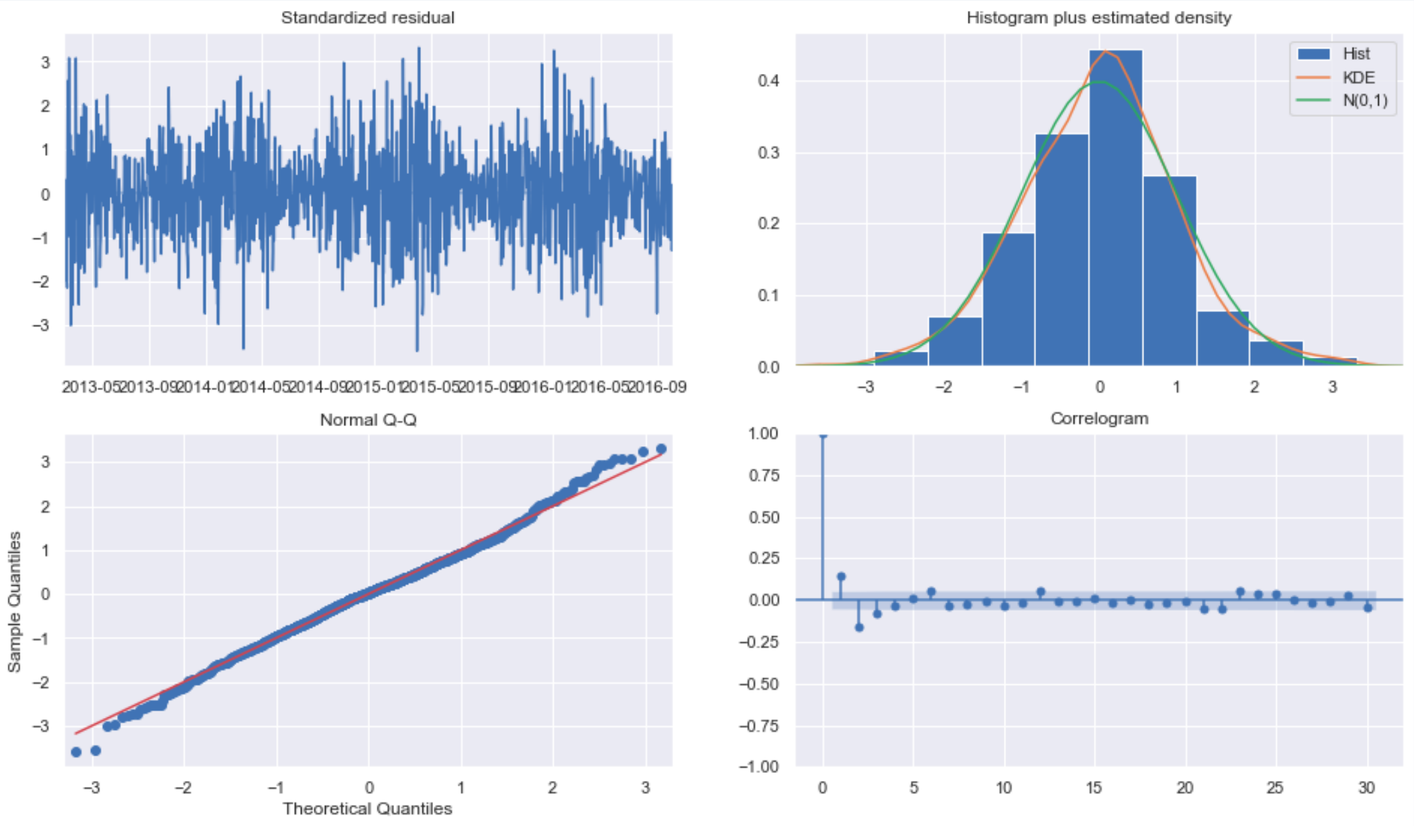
AR(1), Lag=7: the training data is able to match the original data well. However, the test data prediction does not reflect the actual data in the test dataframe, with a very large 95% confidence interval



AR(1), Lag=365: the training data is able to match the original data well. The test data prediction is able to predict actual data in the test dataframe in much greater detail and a smaller 95% confidence interval. Thus, I conclude that this is the better AutoRegression model.

Step 5 - Evaluate the AutoRegression Model

...



A model with a good fit should have residuals that are uncorrelated with zero means.

It can be seen that the model results in zero-mean residuals.

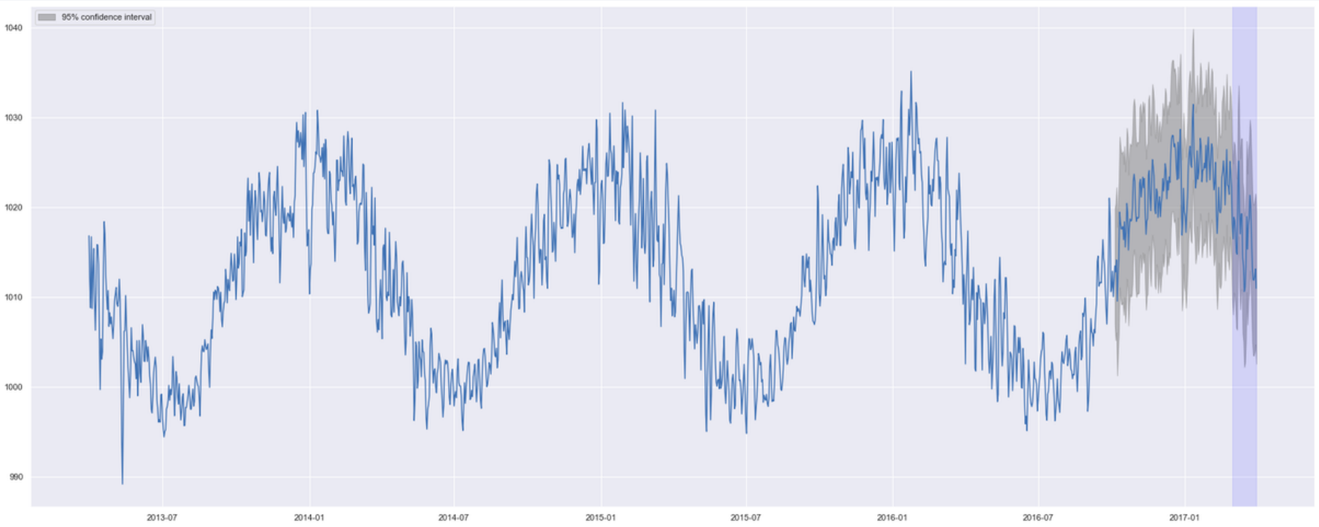
By plotting the histogram of the residuals and the quantile-quantile plot, it can be seen that the residuals are indeed normally distributed.

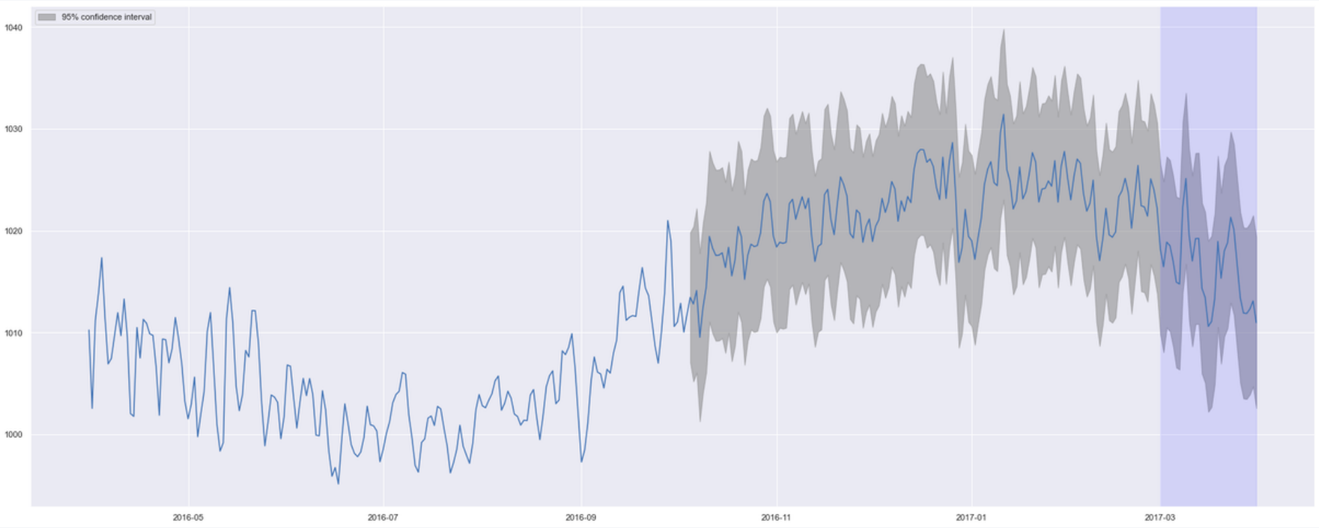
Lastly, the residuals have a sharp cutoff after lag = 0, suggesting that there are little to no correlation between the residuals

For all of these reasons, I can conclude that **AR(1), Lag = 365** is a good fit for making the time series forecast here.

Step 6 - Make a forecast with the AutoRegression Model

...





The forecast of the PRES from March 1, 2017 to March 31, 2017 (highlighted purple) can then be made with the model as shown.

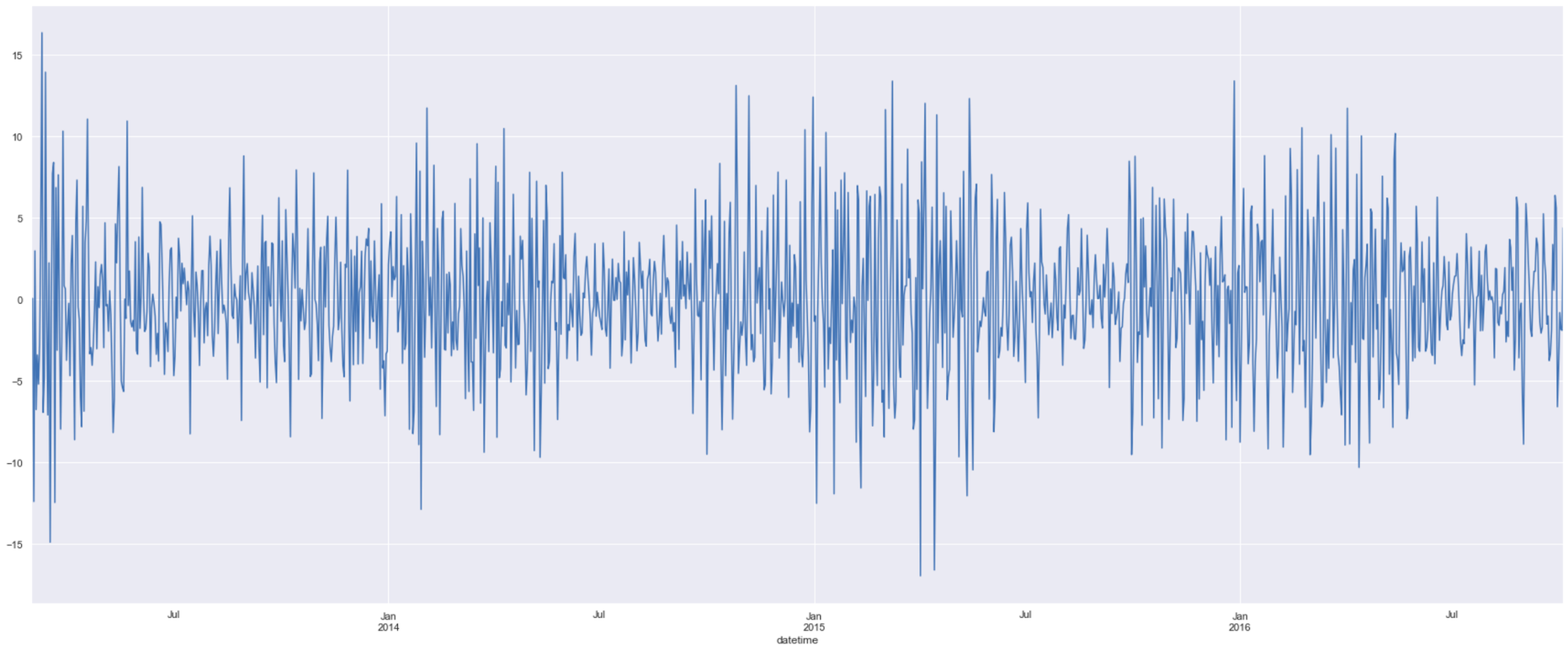
The grey range represents the 95% confidence interval of the forecast

Step 7 - Build the ARIMA Model

...

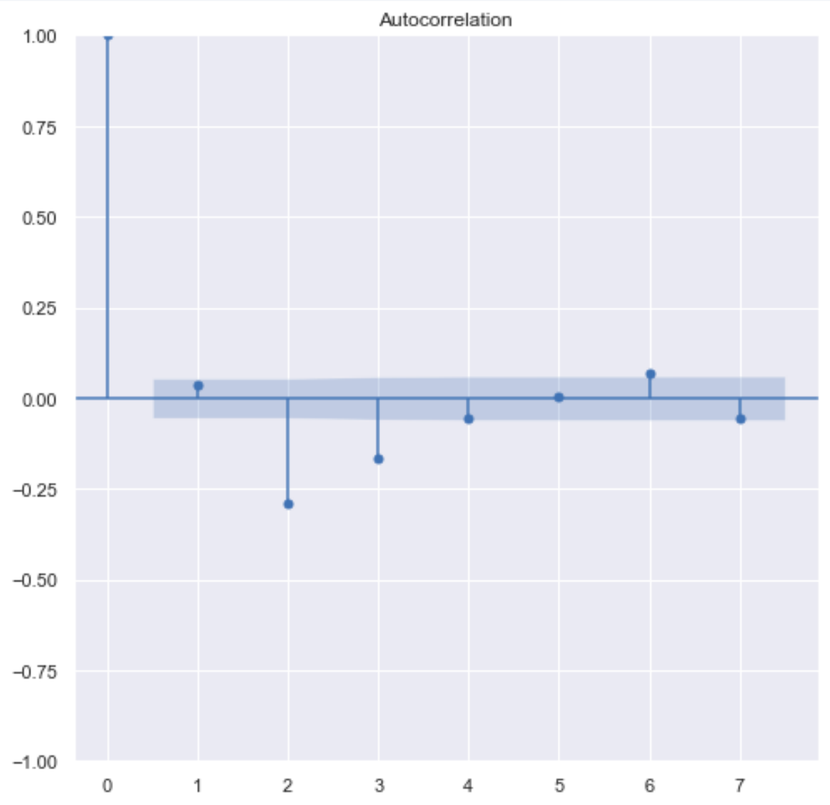
The ARIMA model can be used to model the stationary time series. However, it can be seen that the original data contains seasonality, meaning that we have to eliminate it in order to make the data stationary. This can be achieved by differencing (diff)

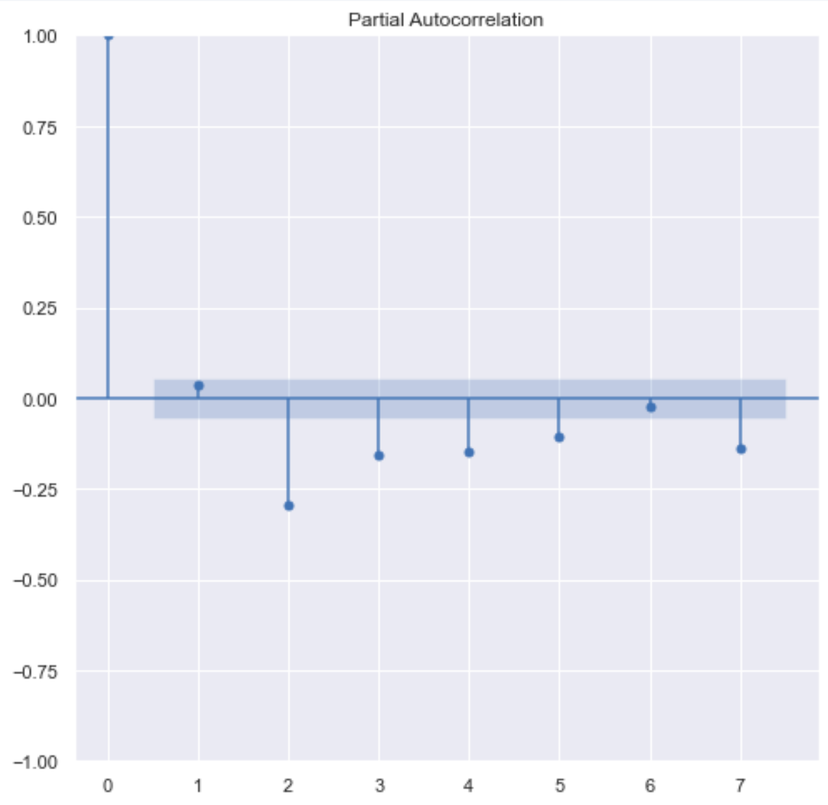
Using seasonal_periods=7 and k_diff=1, the result is as follow:



Step 7 - Build the SARIMA Model

...





The ACF plot shows a cut-off behavior at lags 0, 3, and 6. These will be the values that should be examined for the q value of the MA model

The PACF plot shows the cut-off behavior at lags 0 and 5. These will be the values that should be examined for the p value of the AR model

Step 8 - Evaluating the SARIMA Models

...

SARIMA (2, 1, 2) x (1, 1, 0, 7)

SARIMAX Results

Dep. Variable:	PRES	No. Observations:	1461
Model:	ARIMA(2, 1, 2)x(1, 1, [], 7)	Log Likelihood	-4308.590
Date:	Wed, 15 Feb 2023	AIC	8629.180
Time:	14:13:36	BIC	8660.868
Sample:	03-01-2013	HQIC	8641.004
	- 02-28-2017		

SARIMA (5, 1, 3) x (1, 1, 0, 7)

SARIMAX Results

Dep. Variable:	PRES	No. Observations:	1461
Model:	ARIMA(5, 1, 3)x(1, 1, [], 7)	Log Likelihood	-4232.901
Date:	Wed, 15 Feb 2023	AIC	8485.801
Time:	14:20:18	BIC	8538.615
Sample:	03-01-2013	HQIC	8505.507
	- 02-28-2017		

SARIMA (4, 1, 3) x (1, 1, 0, 7)

SARIMAX Results

Dep. Variable:	PRES	No. Observations:	1461
Model:	ARIMA(4, 1, 3)x(1, 1, [], 7)	Log Likelihood	-4232.901
Date:	Wed, 15 Feb 2023	AIC	8483.801
Time:	13:56:52	BIC	8531.334
Sample:	03-01-2013	HQIC	8501.537
	- 02-28-2017		

SARIMA (4, 1, 3) x (1, 1, 0, 7) is the best-performing model with the lowest AIC and BIC values

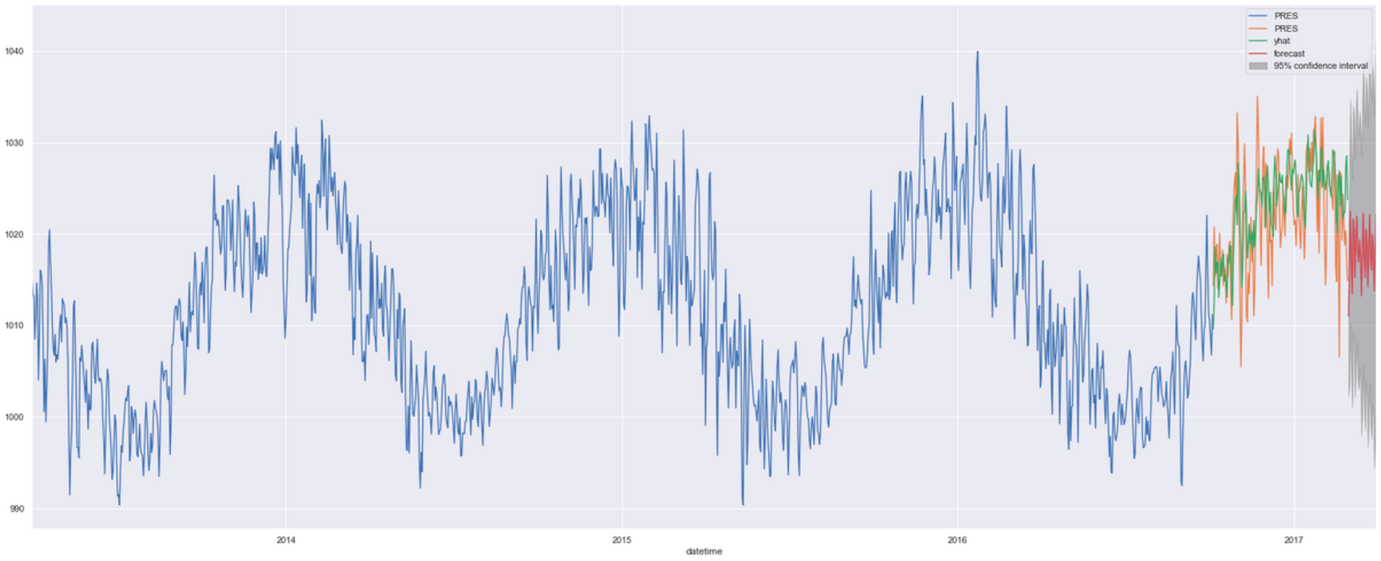
Step 8 - Evaluating the SARIMA Models

...

Evaluating the SARIMA (4, 1, 3) x (1, 1, 0, 7) model against the test data reveals that the SARIMA model (Green) is actually a pretty good fit compared to the Actual Data (Blue) and AutoRegression Prediction (Orange)

Step 9 - Make a forecast with the SARIMA Model

...



SARIMA (4, 1, 3) x (1, 1, 0, 7)

SARIMAX Results

Dep. Variable:	PRES	No. Observations:	1461
Model:	ARIMA(4, 1, 3)x(1, 1, [], 7)	Log Likelihood	-4232.901
Date:	Wed, 15 Feb 2023	AIC	8483.801
Time:	13:56:52	BIC	8531.334
Sample:	03-01-2013	HQIC	8501.537
	- 02-28-2017		

SARIMA (4, 1, 3) x (1, 1, 0, 7): The forecast for daily pressure in March is represented by the red line and the 95% confidence interval is reflected as the grey region.

This combination of p, d, q, P, D, Q, s values yields the lowest AIC and BIC values so far.

The resulting forecast shows a slight decrease in pressure, similar to the results in the previous years.

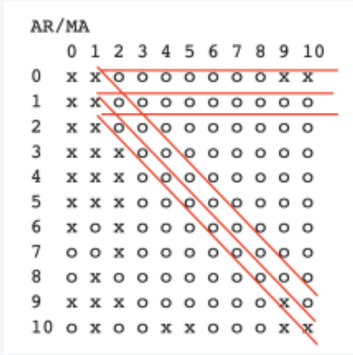
Thus, I conclude that the model is decent for predicting the March 2017 daily pressure.

Step 9 - Improving the Models

AR(1), Lag = 7 and SARIMA (4, 1, 3) x (1, 1, 0, 7) are the models deemed suitable for making forecasts of the provided pressure data in Tiantan, March 2017.

While the results are satisfactory, some improvements can be made to further improve the model:

I can use R programming language to determine the EACF diagram, which could help identify the appropriate p and q variables used to fit the ARIMA model



Example of EACF table

Once time has passed and there are more data available, the model could be updated with the newer data, to forecast even further longer future pressure values.