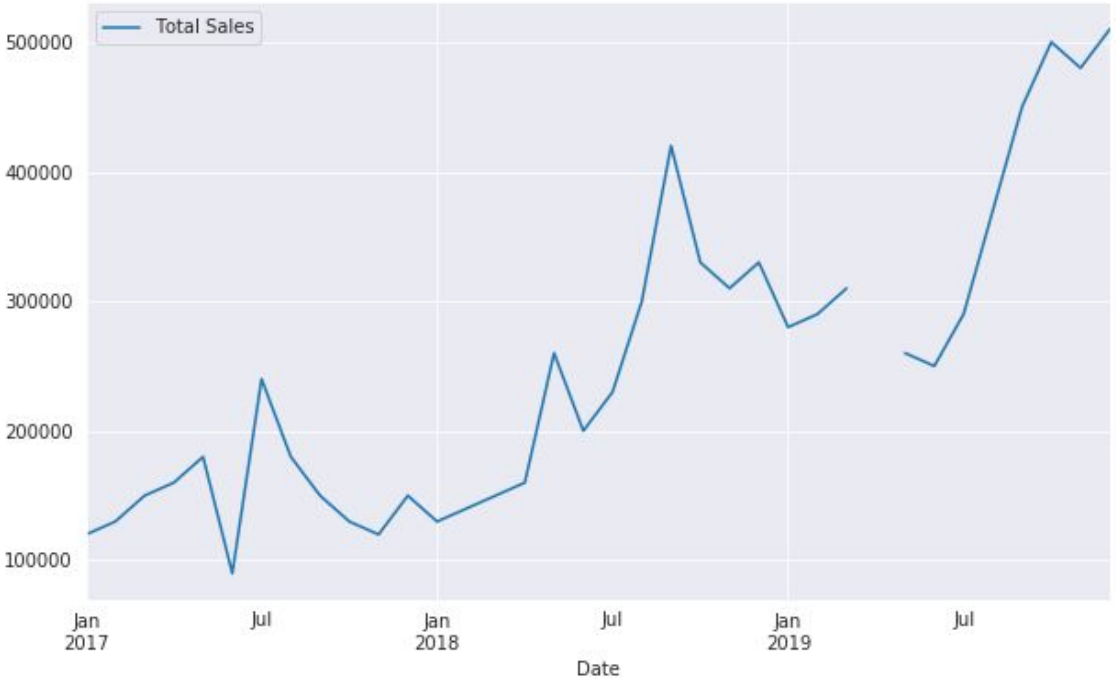
First AI & DS bootcamp in Egypt

Team 1 project
Time series Analysis

Used Libraries:

```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, mean_absolute_error
# plt.style.use('fivethirtyeight') # For plots
sns.set_style("darkgrid")
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 10, 6
import os
```

Sales graph before cleaning



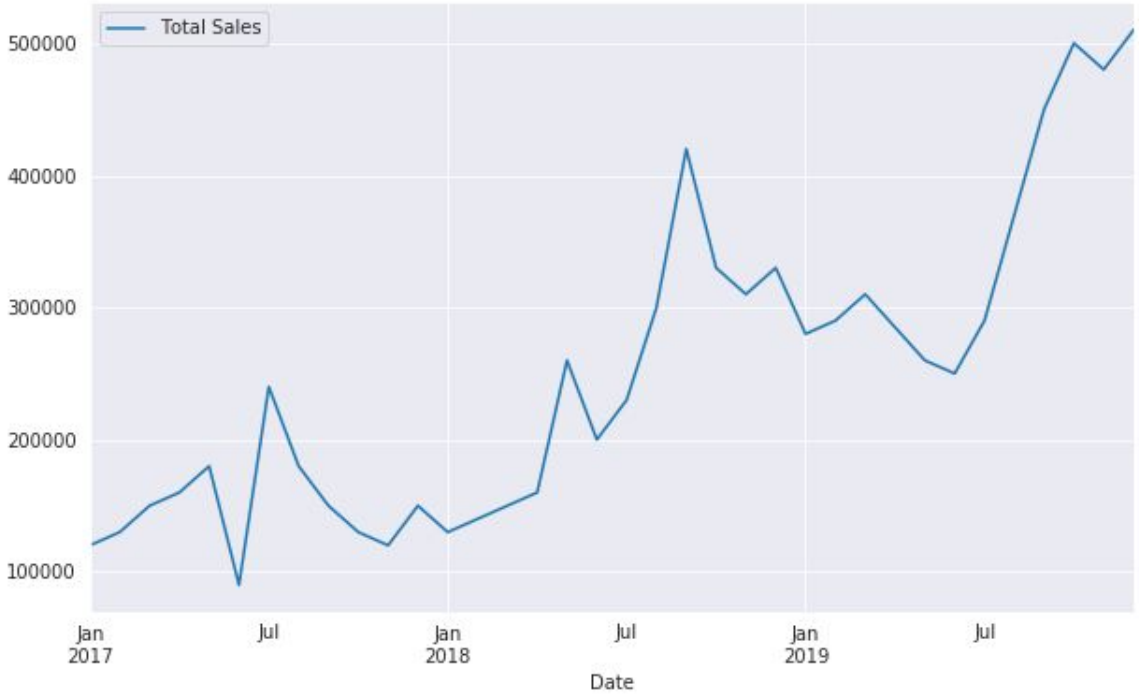
Null value results in an blank spot in the graph

Cleaning the data

```
[23] #find the NaNs locations
     missIndex = sales['Total_Sales'].index[sales['Total_Sales'].apply(np.isnan)]
     print(missIndex)
```

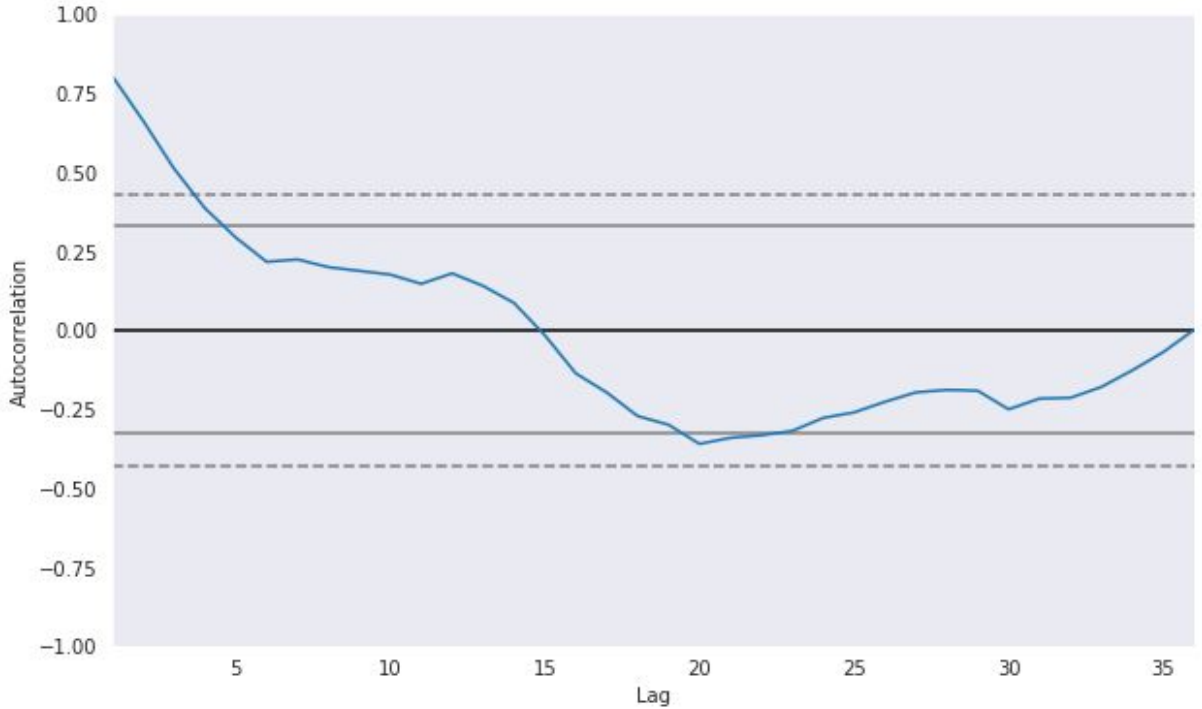
```
[24] #get the average to replace NaN
     sales.loc['2019-04-01'] = (sales.loc['2019-03-01'] + sales.loc['2019-05-01'])/2

     print(sales.loc['2019-04-01'])
```

Graph after replacing null value



Graph after replacing null value with average

Autocorrelation plot



Training the model

```
[28] from statsmodels.tsa.arima_model import ARIMA

     # fit model
     model = ARIMA(sales, order=(5,1,0))
     model_fit = model.fit(disp=0)
     print(model_fit.summary())
```

Results

ARIMA Model Results

Dep. Variable:	D.Total Sales	No. Observations:	35
Model:	ARIMA(5, 1, 0)	Log Likelihood	-429.618
Method:	css-mle	S.D. of innovations	51791.202
Date:	Mon, 03 Feb 2020	AIC	873.236
Time:	14:53:44	BIC	884.124
Sample:	02-01-2017 - 12-01-2019	HQIC	876.994

	coef	std err	z	P>|z|	[0.025	0.975]
const	1.119e+04	6927.188	1.615	0.117	-2388.416	2.48e+04
ar.L1.D.Total Sales	-0.2219	0.167	-1.328	0.195	-0.549	0.106
ar.L2.D.Total Sales	-0.0703	0.169	-0.417	0.680	-0.401	0.260
ar.L3.D.Total Sales	-0.0280	0.170	-0.165	0.870	-0.362	0.306
ar.L4.D.Total Sales	0.0384	0.176	0.218	0.829	-0.307	0.384
ar.L5.D.Total Sales	0.0037	0.182	0.020	0.984	-0.353	0.360

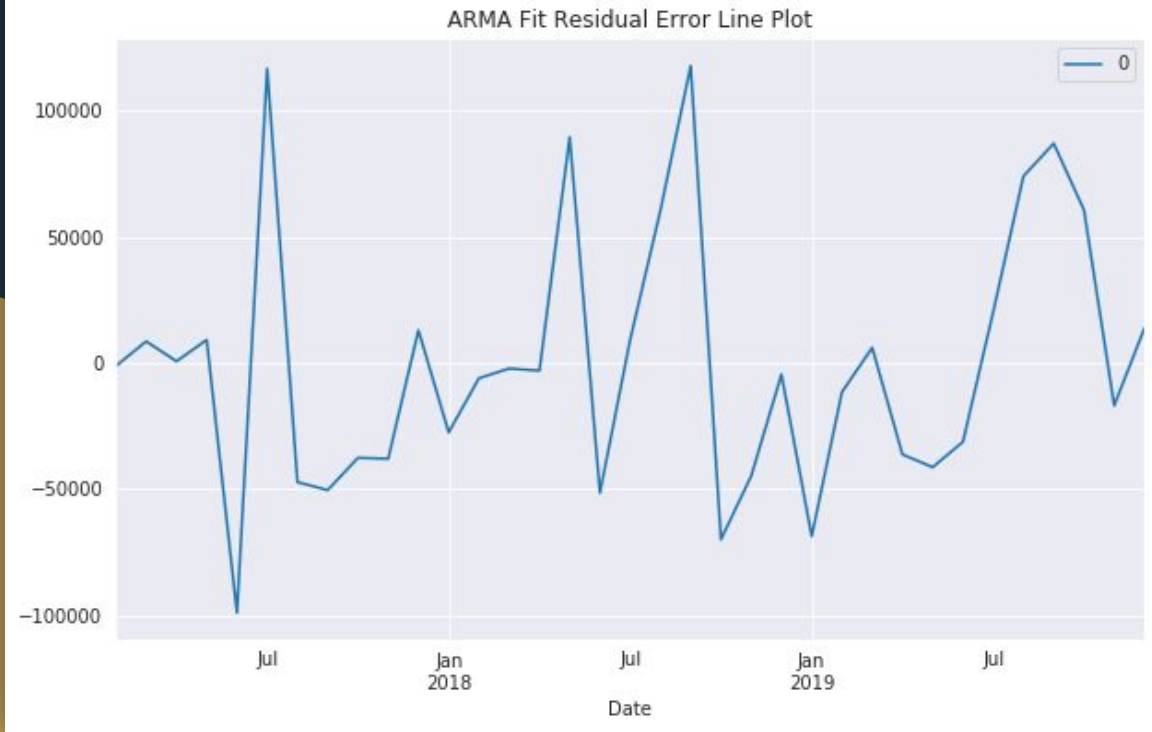
Roots

	Real	Imaginary	Modulus	Frequency
AR.1	-2.0930	-0.0000j	2.0930	-0.5000
AR.2	-0.0393	-2.0790j	2.0794	-0.2530
AR.3	-0.0393	+2.0790j	2.0794	0.2530
AR.4	2.7306	-0.0000j	2.7306	-0.0000
AR.5	-10.8482	-0.0000j	10.8482	-0.5000

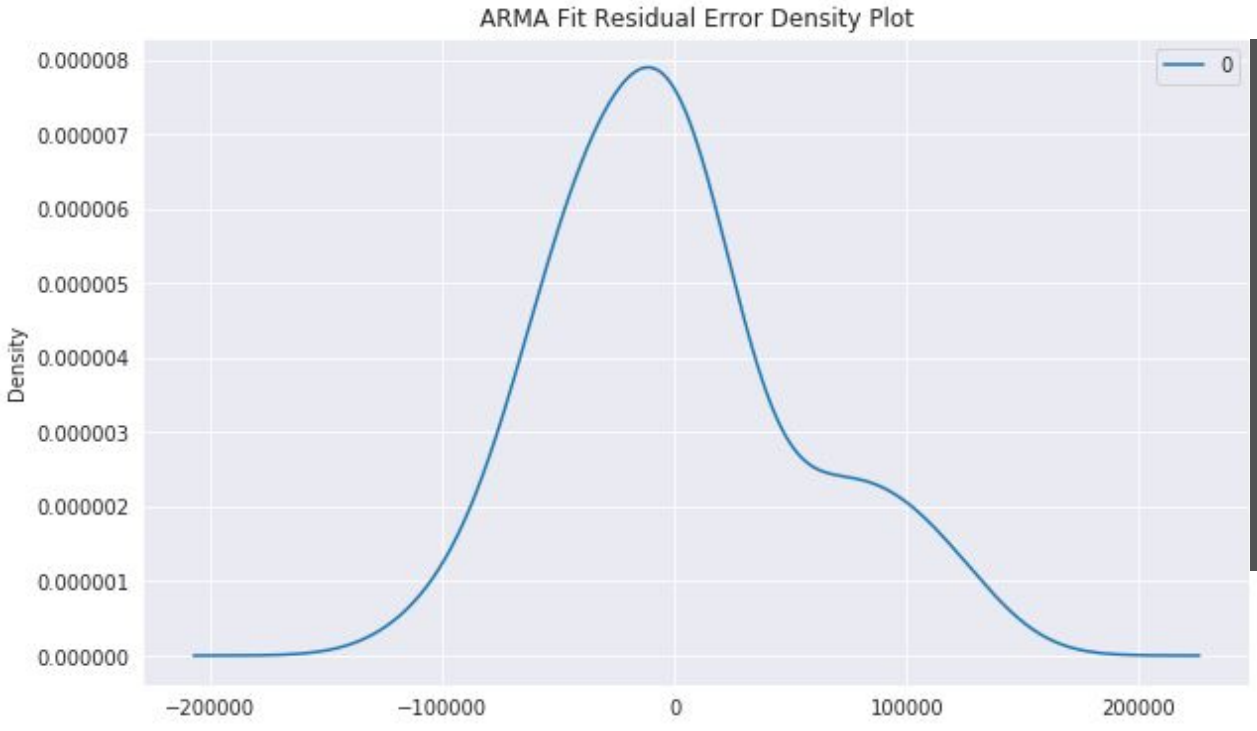
Plotting the results

```
# plot residual errors
residuals = pd.DataFrame(model_fit.resid)
residuals.plot()
plt.title('ARMA Fit Residual Error Line Plot')
plt.show()

residuals.plot(kind='kde')
plt.title('ARMA Fit Residual Error Density Plot')
plt.show()
print(residuals.describe())
```

Results plot

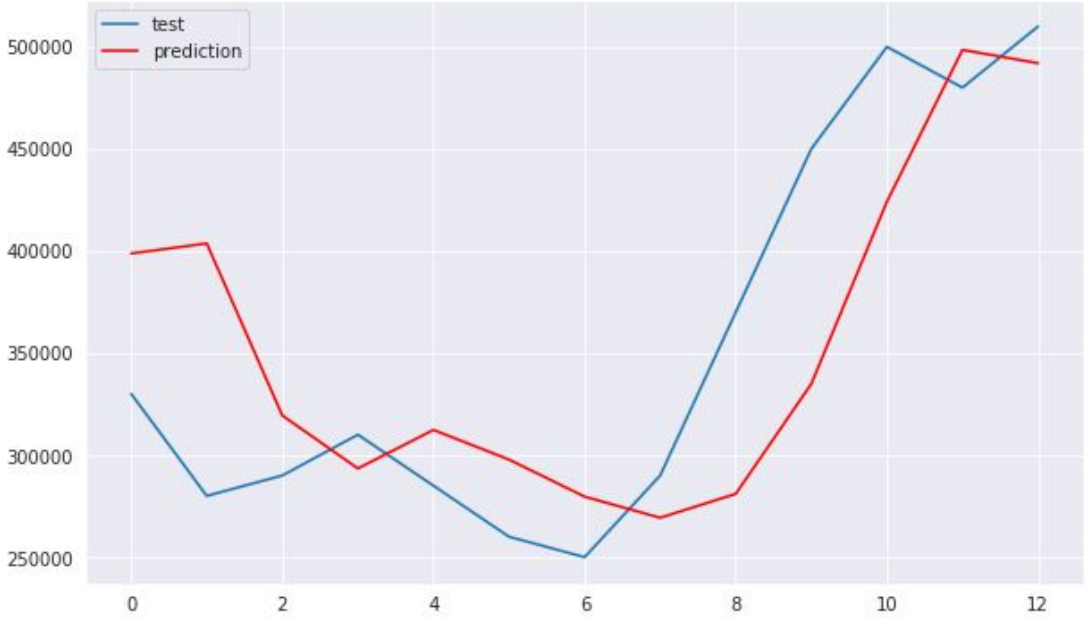


Results plot



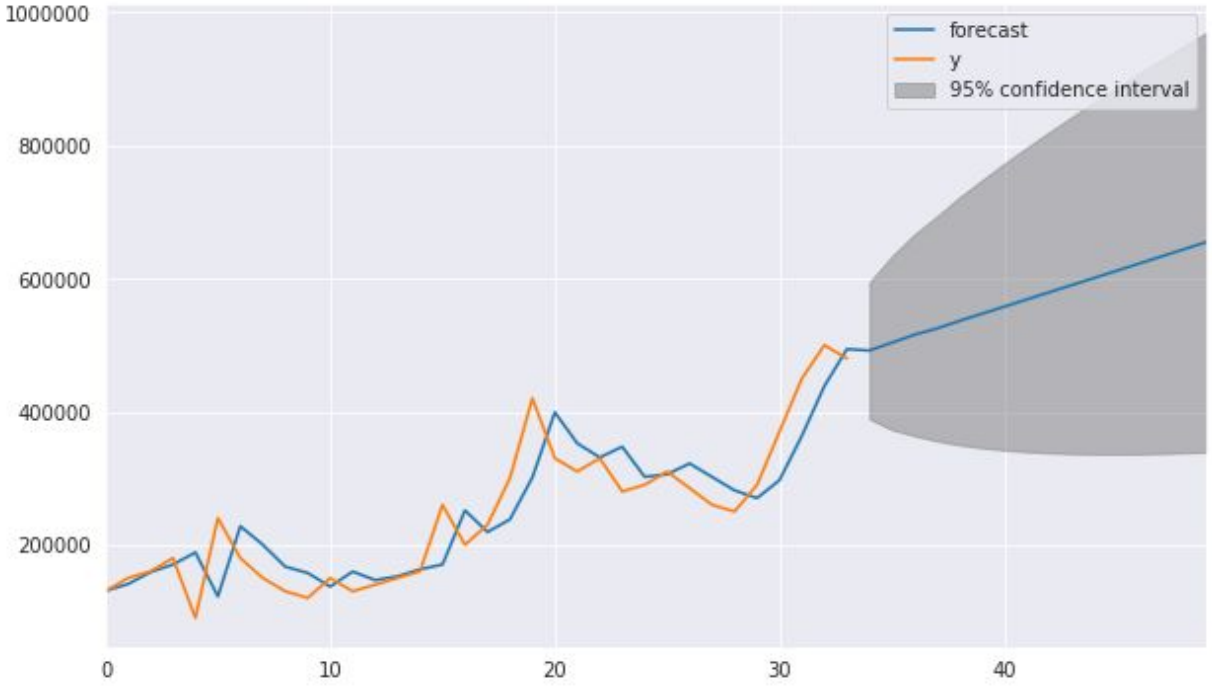
	0
count	35.000000
mean	-51.358917
std	52547.526967
min	-99024.540897
25%	-37771.575702
50%	-3052.052644
75%	13387.377625
max	117787.287681

Testing the model




```
predicted=398769.926486, expected=330000.000000
predicted=403667.017026, expected=280000.000000
predicted=319385.073195, expected=290000.000000
predicted=293426.564548, expected=310000.000000
predicted=312325.645595, expected=285000.000000
predicted=297785.041851, expected=260000.000000
predicted=279631.905304, expected=250000.000000
predicted=269272.995995, expected=290000.000000
predicted=281006.127973, expected=370000.000000
predicted=334879.828889, expected=450000.000000
predicted=424110.778996, expected=500000.000000
predicted=498502.447426, expected=480000.000000
predicted=491986.769570, expected=510000.000000
Test MSE: 4018585225.464
```

Forecasting



Questions.