```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.arima_model import ARIMA
import datetime
import itertools
import warnings
from sklearn.metrics import mean_squared_error
import seaborn as sns
import statsmodels
plt.style.use('fivethirtyeight')
%matplotlib inline
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning:
  import pandas.util.testing as tm
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
df = pd.read_csv("/content/drive/MyDrive/Ineuron/data_stocks.csv")
df.head()
```

	DATE	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP
0	1491226200	2363.6101	42.3300	143.6800	129.6300	82.040	102.2300
1	1491226260	2364.1001	42.3600	143.7000	130.3200	82.080	102.1400
2	1491226320	2362.6799	42.3100	143.6901	130.2250	82.030	102.2125
3	1491226380	2364.3101	42.3700	143.6400	130.0729	82.000	102.1400
4	1491226440	2364.8501	42.5378	143.6600	129.8800	82.035	102.0600

5 rows × 502 columns

```
df["DATE"].dtypes
```

```
dtype('int64')
```

```
df['DATE'] = pd.to_datetime(df['DATE'], unit='s')
```

```
df['DATE'].tail()
```

```
41261   2017-08-31 19:56:00
41262   2017-08-31 19:57:00
41263   2017-08-31 19:58:00
41264   2017-08-31 19:59:00
41265   2017-08-31 20:00:00
Name: DATE, dtype: datetime64[ns]
```

```
df.index = df['DATE']
```

```
df.drop('DATE',axis = 1,inplace=True)
```

```
df.tail()
```

	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.A
DATE							
2017-08-31 19:56:00	2472.22	44.72	164.11	155.090	83.67	106.565	11
2017-08-31 19:57:00	2471.77	44.73	164.12	155.160	83.65	106.590	11
2017-08-31 19:58:00	2470.03	44.74	164.01	155.065	83.62	106.520	11
2017-08-31 19:59:00	2471.49	44.71	163.88	154.960	83.58	106.400	11
2017-08-31 20:00:00	2471.49	44.74	163.98	155.160	83.69	106.470	11

5 rows × 501 columns

NASDAQ.ADP

```
df_ADP = df[['NASDAQ.ADP']].copy()
df_ADP.tail()
```

	NASDAQ.ADP
DATE	
2017-08-31 19:56:00	106.565
2017-08-31 19:57:00	106.590

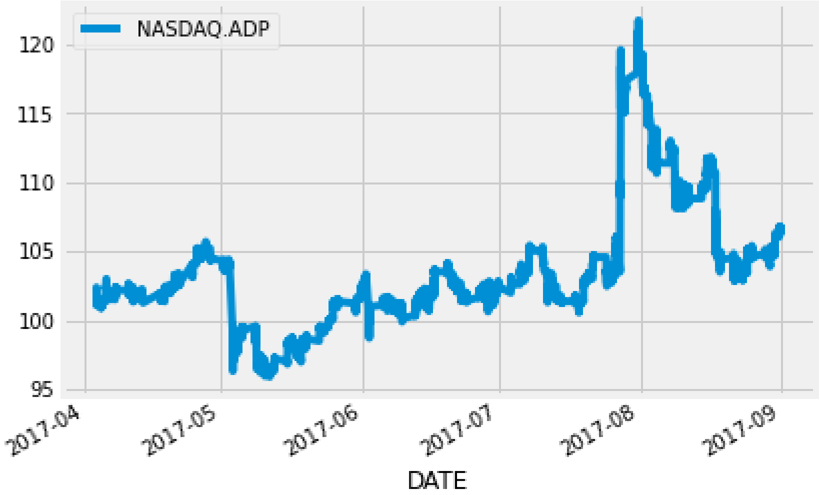
```
df_ADP.count()
```

```
NASDAQ.ADP    41266
dtype: int64
```

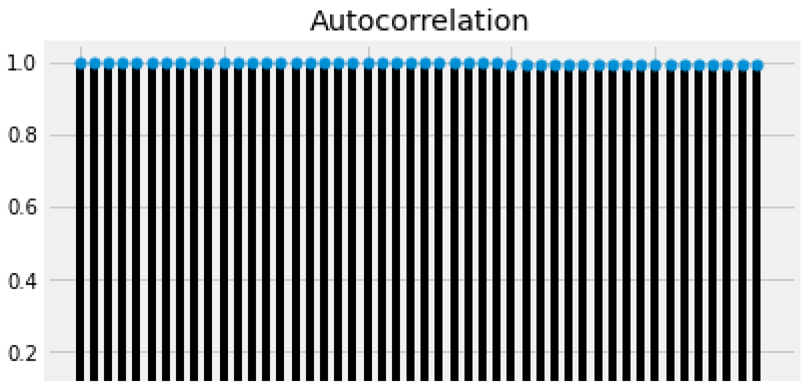
2017-08-31 20:00:00	106.470

```
df_ADP.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f96167444a8>
```



```
from statsmodels.graphics.tsaplots import plot_acf
plot_acf(df_ADP)
```



```
##Converting series to stationary
df_ADP.shift(1)
```

DATE	NASDAQ.ADP
2017-04-03 13:30:00	NaN
2017-04-03 13:31:00	102.2300
2017-04-03 13:32:00	102.1400
2017-04-03 13:33:00	102.2125
2017-04-03 13:34:00	102.1400
...	...
2017-08-31 19:56:00	106.6300
2017-08-31 19:57:00	106.5650
2017-08-31 19:58:00	106.5900
2017-08-31 19:59:00	106.5200
2017-08-31 20:00:00	106.4000

41266 rows × 1 columns

```
X = df_ADP.values
train = X[0:28886] # 27 data as train data
test = X[28886:] # 9 data as test data
print(train.size)
print(test.size)
predictions = []
```

```
28886
12380
```

```
p=d=q=range(0,2)
pdq=list(itertools.product(p,d,q))

warnings.filterwarnings('ignore')
warnings.filterwarnings('ignore')
for param in pdq:
    try:
        model_arima = ARIMA(train, order=param)
        model_arima_fit = model_arima.fit()
        print(param,model_arima_fit.aic)
    except:
        continue
```

```
     (0, 0, 0) 124317.93290534396
     (0, 0, 1) 85271.48908067068
     (0, 1, 0) -80762.52187440016
     (0, 1, 1) -81075.63405539667
     (1, 0, 0) -80762.97956376115
     (1, 0, 1) -81077.32276388479
     (1, 1, 0) -81067.89180999548
     (1, 1, 1) -81073.99649148404
```

```
from statsmodels.tsa.arima_model import ARIMA
model_arima = ARIMA(train, order=(2,1,2))
model_arima_fit = model_arima.fit()
```

```
#p,d,q
#p -> Periods taken for auto regressive model
#d -> Integrated order, difference
#q -> Periods in moving average model
from statsmodels.tsa.arima_model import ARIMA
model_arima = ARIMA(train, order=(3,1,3))
model_arima_fit = model_arima.fit()
print(model_arima_fit.aic)
```

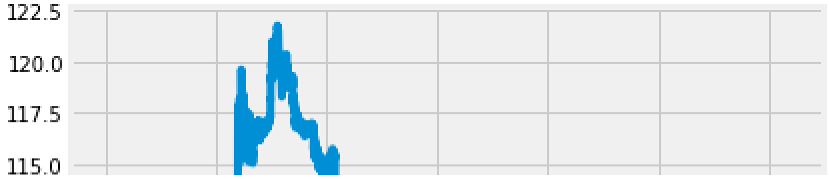
```
     -81073.99649148404
```

```
predictions = model_arima_fit.forecast(steps=12380)[0]
predictions
```

```
     array([102.67200781, 102.67196666, 102.67198318, ..., 102.85692098,
            102.85693592, 102.85695086])
```

```
plt.plot(test)
plt.plot(predictions, color='red')
```

```
[<matplotlib.lines.Line2D at 0x7f960d6d5fd0>]
```



```
res =round(mean_squared_error(test,predictions),2)
```

```
res
```

```
52.8
```