DSBDA LAB 6

RAVITEJ.R
160119733160

AIM:- To do time series analysis using ARIMA on a dataset

DESCRIPTION:-

In the domain of machine learning, there's a specific collection of methods and techniques particularly well suited for predicting the value of a dependent variable according to time. In the proceeding article, we'll cover AutoRegressive Integrated Moving Average (ARIMA).

We refer to a series of data points indexed (or graphed) in time order as a time series. A time series can be broken down into 3 components.

- Trend: Upward & downward movement of the data with time over a large period of time (i.e. house appreciation)
- Seasonality: Seasonal variance (i.e. an increase in demand for ice cream during summer)
- Noise: Spikes & troughs at random intervals

Augmented Dickey-Fuller Test: The time series is considered stationary if the p-value is low (according to the null hypothesis) and the critical values at 1%, 5%, 10% confidence intervals are as close as possible to the ADF Statistics

AutoRegressive Integrated Moving Average Model (ARIMA)
The ARIMA (aka Box-Jenkins) model adds differencing to an ARMA model. Differencing subtracts the current value from the previous and can be used to transform a time series into one that's stationary. For example, first-order differencing addresses linear trends, and employs the transformation zi = yi — yi-1. Second-order differencing addresses quadratic trends and employs a first-order difference of a first-order difference, namely zi = (yi — yi-1) — (yi-1 — yi-2), and so on.

Three integers (p, d, q) are typically used to parametrize ARIMA models.

p: number of autoregressive terms (AR order)
d: number of nonseasonal differences (differencing order)

q: number of moving-average terms (MA order)

Auto Correlation Function (ACF)
The correlation between the observations at the current point in time and the observations at all previous points in time. We can use ACF to determine the optimal number of MA terms. The number of terms determines the order of the model.

Partial Auto Correlation Function (PACF)
As the name implies, PACF is a subset of ACF. PACF expresses the correlation between observations made at two points in time while accounting for any influence from other data points. We can use PACF to determine the optimal number of terms to use in the AR model. The number of terms determines the order of the model.

CODE:-

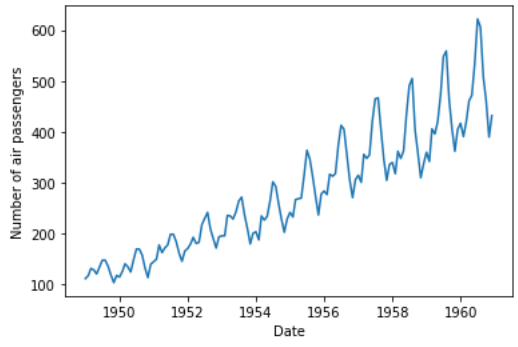
```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.arima_model import ARIMA
from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()
# reading the given dataset
df = pd.read_csv('AirPassengers.csv', parse_dates = ['Month'], index_col = ['Month'])
df.head()
print(df)
plt.xlabel('Date')
plt.ylabel('Number of air passengers')
plt.plot(df)
# performing ADF test to see if data set is stationary
result = adfuller(df['#Passengers'])
print('ADF Statistic: {}'.format(result[0]))
print('p-value: {}'.format(result[1]))
print('Critical Values:')
for key, value in result[4].items():
    print('\t{}: {}'.format(key, value))
# applying log as test failed
df_log = np.log(df)
plt.plot(df_log)
```

```
result = adfuller(df_log['#Passengers'])
print('ADF Statistic: {}'.format(result[0]))
print('p-value: {}'.format(result[1]))
print('Critical Values:')
for key, value in result[4].items():
    print('\t{}: {}'.format(key, value))
#test passed hence applying arima to the data set
model = ARIMA(df_log, order=(2,1,2))
results = model.fit(disp=-1)
plt.plot(results.fittedvalues, color='red')
# predicting the future traffic using models result
results.plot_predict(1,264)
```

OUTPUT:-

```
            #Passengers
Month
1949-01-01          112
1949-02-01          118
1949-03-01          132
1949-04-01          129
1949-05-01          121
...                 ...
1960-08-01          606
1960-09-01          508
1960-10-01          461
1960-11-01          390
1960-12-01          432

[144 rows x 1 columns]
[<matplotlib.lines.Line2D at 0x7fa720da5d90>]
```



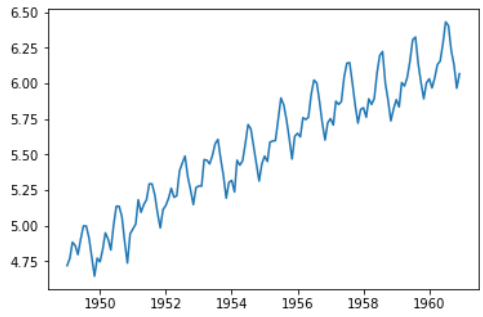
Reading the dataset and plotting for analysis

```
ADF Statistic: 0.8153688792060472
p-value: 0.991880243437641
Critical Values:
        1%: -3.4816817173418295
        5%: -2.8840418343195267
        10%: -2.578770059171598
```

APPLYING ADF test to see if the dataset is stationary it fails

```
plt.plot(df_log)
```

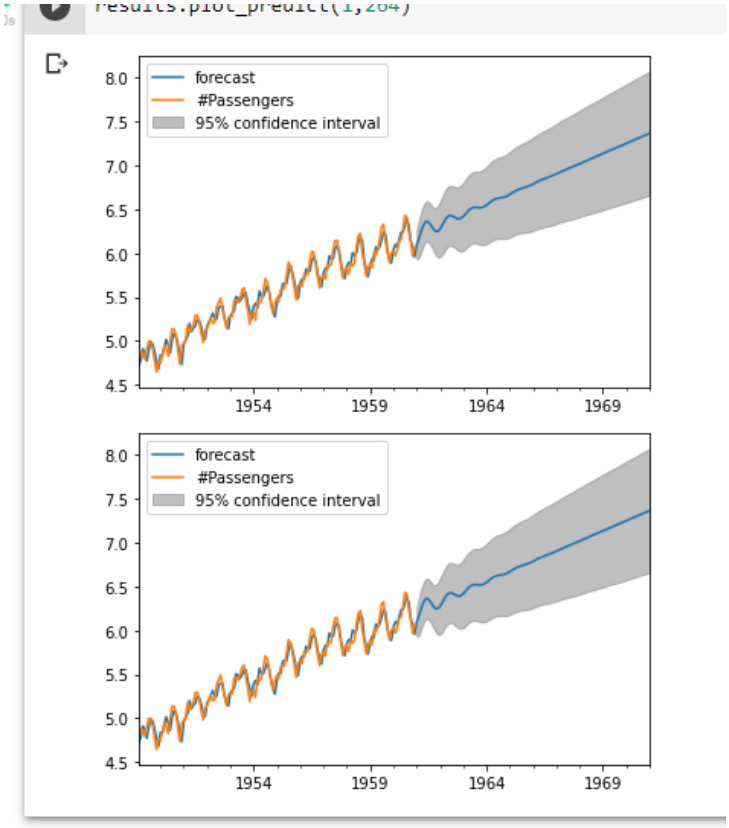
```
[<matplotlib.lines.Line2D at 0x7fa720d80150>]
```




```
        print('\t{}: {}'.format(key, value))
```

```
ADF Statistic: -1.7170170891069603
p-value: 0.4223667747703914
Critical Values:
        1%: -3.4816817173418295
        5%: -2.8840418343195267
        10%: -2.578770059171598
```

Applying log to the data as to decrease the pvalue i.e reducing the time component and redoing the test it passes



PREDICTED RESULTS