Terna Engineering College

Department of Artificial Intelligence and Data Science

Program: Sem VI

Course: Data Analytics and Visualization Lab

Experiment No.06

PART A

(PART A: TO BE REFERRED BY STUDENTS)

A.1 Aim: To implement Time Series Analysis in Python/R.

A.2 Theory:

Time Series Analysis:

A time series is an ordered sequence of equally spaced values over time or a sequence of data points that occur in successive order over some period of time. For e.g. In investing, it is common to use a time series to track the price of a security over time. This can be tracked over the short term, such as the price of a security on the hour over the course of a business day, or the long term, such as the price of a security at close on the last day of every month over the course of five years.

Time series analysis has 2 main goals or objectives:

- Identify and model the structure of the time series.
- Forecast future values in the time series.

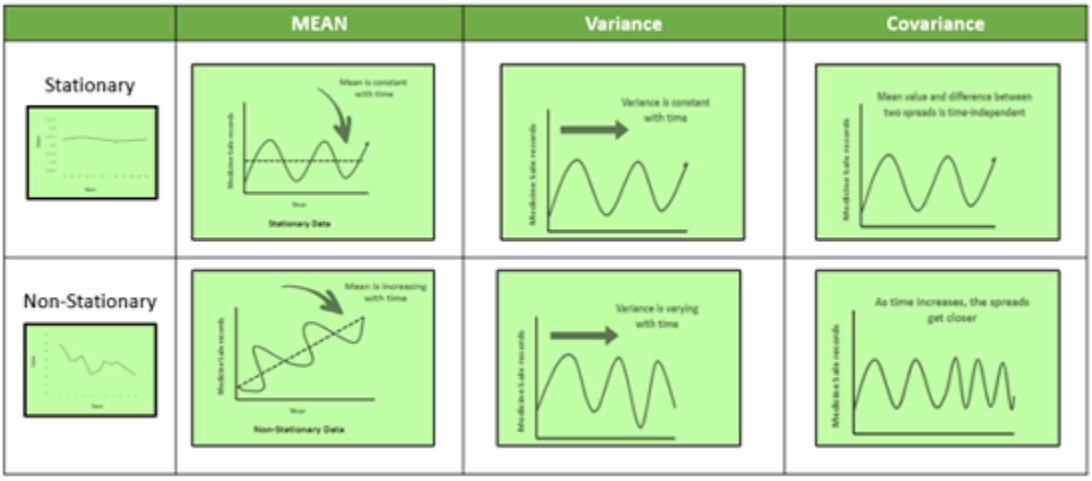
Types of time series:

Stationary: A dataset should follow the below thumb rules without having Trend, Seasonality, Cyclical, and Irregularity components of the time series.

- The mean value of them should be completely constant in the data during the analysis.

- The variance should be constant with respect to the time-frame
- Covariance measures the relationship between two variables.

Non-stationary: If either the mean-variance or covariance is changing with respect to time, the dataset is called non-stationary.



Designed by Author (Shanthababu)

Forecasting in time series:

Forecasting involves taking models fit on historical data and using them to predict future observations. There are different approaches to predict the value, consider an example there is a company XYZ records the website traffic in each hour and now wants to forecast the total traffic of the coming hour.

While forecasting time series values, 3 important terms need to be taken care of and the main task of time series forecasting is to forecast these three terms:

1. Seasonality: It is a simple term that means while predicting a time series data there are some points in a particular domain where the output value is at a peak as compared to other points.
2. Trend: describes that there is certainly increasing or decreasing pattern in time series.
3. Unexpected Events: Unexpected events mean some dynamic changes occur in an organization, or in the market which cannot be captured.

PART B

(PART B: TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)

Roll. No. A12	Name: Sufiyan Khan
Class: TE – AI & DS	Batch: A1
Date of Experiment:	Date of Submission:
Grade:	

B.1 Input and Output:

```
from datetime import datetime
import numpy as np
import pandas as pd
import matplotlib.pylab as plt
get_ipython().run_line_magic('matplotlib', 'inline')
from statsmodels.tsa.stattools import adfuller
from sklearn.metrics import mean_squared_error
from matplotlib.pylab import rcParams

rcParams['figure.figsize']=10,6
```

✓ 1.7s

```
location=r"AirPassengers.csv"
df= pd.read_csv(location, encoding='gbk',parse_dates=['Month'],infer_datetime_format=True)
indf=df.set_index(['Month'])
```

✓ 0.0s

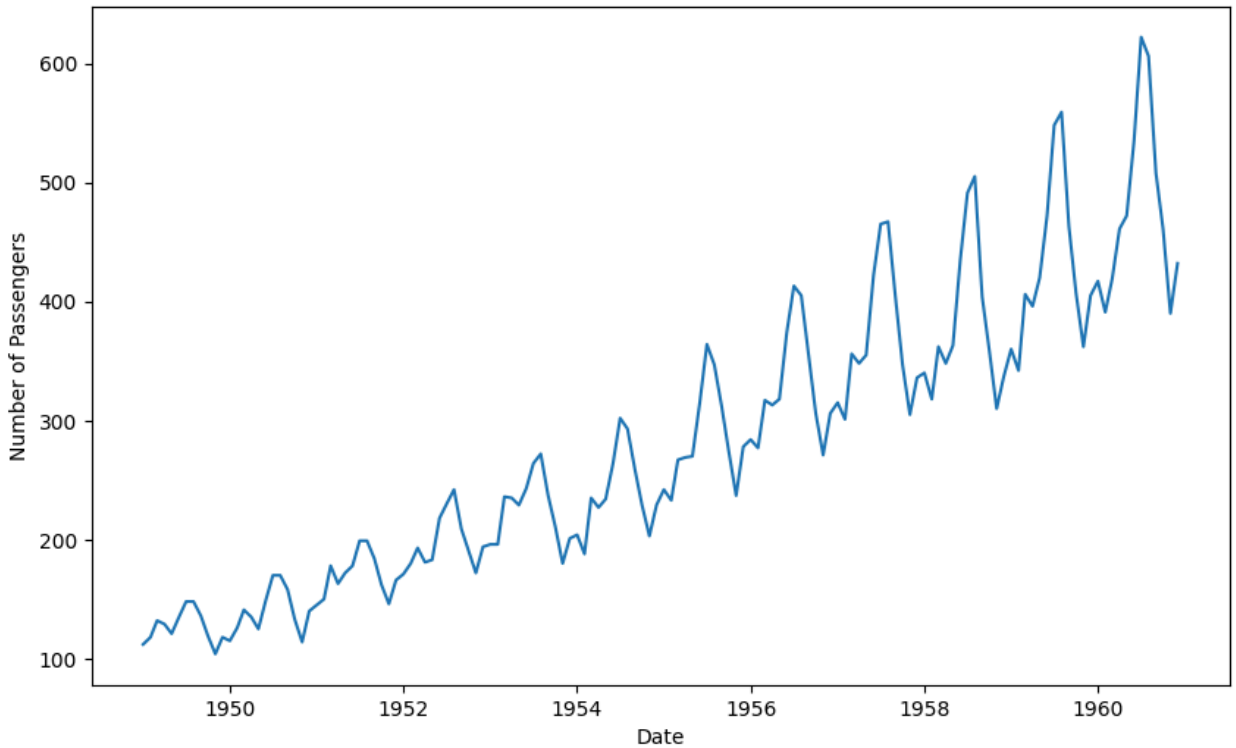
```
indf.describe()
indf.rename(columns={'#Passengers':'Passengers'},inplace=True)
indf.head()
```

✓ 0.0s

Month	Passengers
1949-01-01	112
1949-02-01	118
1949-03-01	132
1949-04-01	129
1949-05-01	121

```
plt.xlabel('Date')
plt.ylabel('Number of Passengers')
plt.plot(indf)
```

✓ 0.2s



```
indf['months'] = [x.month for x in indf.index]
indf['years'] = [x.year for x in indf.index]
```

✓ 0.0s

```
indf.reset_index(drop=True, inplace=True)
```

✓ 0.0s

```
X=indf.drop("Passengers",axis=1)
Y= indf["Passengers"]
X_train=X[:int (len(Y)*0.75)]
X_test=X[int(len(Y)*0.75):]
Y_train=Y[:int (len(Y)*0.75)]
Y_test=Y[int(len(Y)*0.75):]
```

✓ 0.0s

```
from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor()
rf.fit(X_train, Y_train)
```

✓ 0.2s

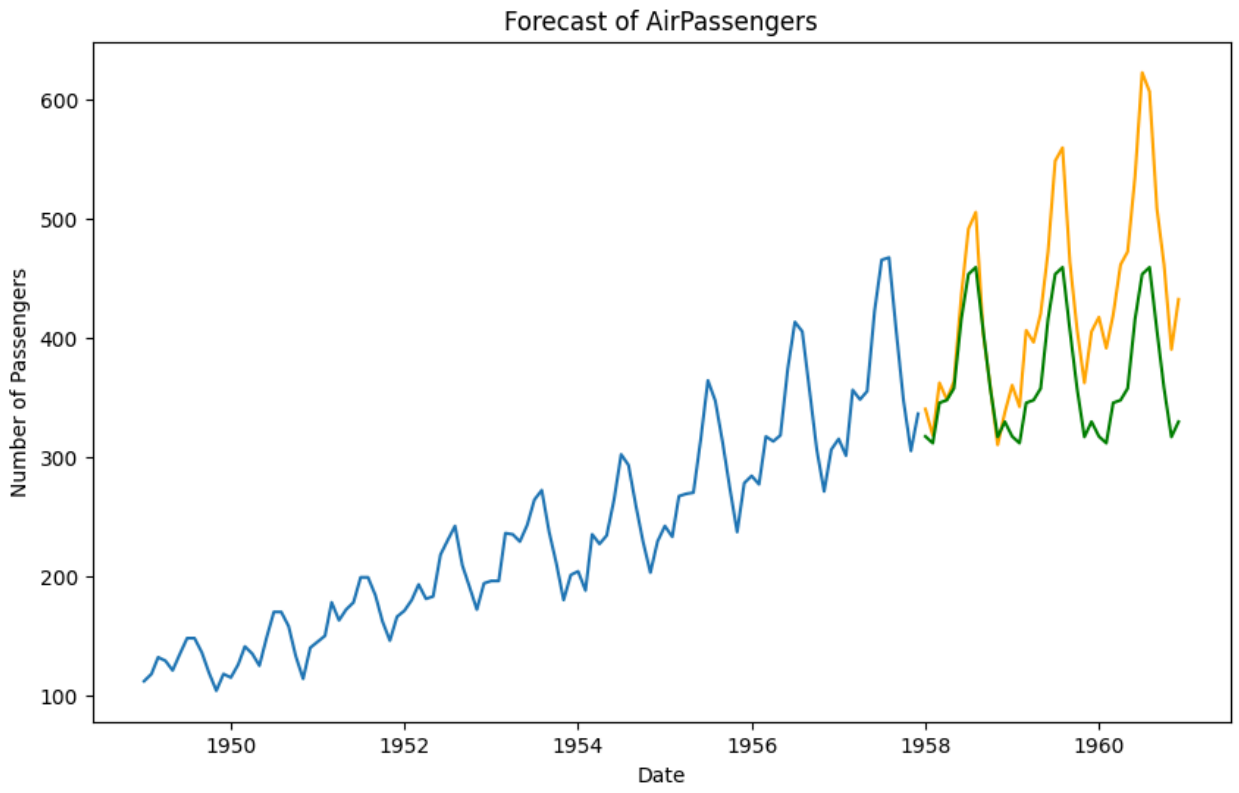
▾ RandomForestRegressor

RandomForestRegressor()

```
df1=df.set_index(['Month'])
df1.rename(columns={'#Passengers':'Passengers'},inplace=True)
train=df1.Passengers[:int (len(indf.Passengers)*0.75)]
test=df1.Passengers[int(len(indf.Passengers)*0.75):]
preds=rf.predict(X_test)
predictions=pd.DataFrame(preds,columns=['Passengers'])
predictions.index=test.index
plt.plot(train)
plt.plot(test, color='orange', label='actual')
plt.plot(predictions,color='green', label='Forecasts')
plt.xlabel('Date')
plt.ylabel('Number of Passengers')
plt.title("Forecast of AirPassengers")
```

✓ 0.2s

```
Text(0.5, 1.0, 'Forecast of AirPassengers')
```




```
print("training score: ", rf.score(X_train, Y_train))
print("testing score: ", rf.score(X_test, Y_test))
```

✓ 0.0s

```
training score:  0.9958006672297484
testing score:  0.07943744546193343
```

B.2 Conclusion:

Thus we have successfully implemented time series analysis in Python on air passengers dataset and understood how the data is distributed and observed the trends in the series. We have also trained the model to forecast future data based on observed trends.