K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Batch: H-DA-8	Roll No.: 16010122284
Experiment No. 6	

TITLE : To perform time series analysis on health care

AIM: To perform forecasting using time series analysis

Expected OUTCOME of Experiment:
CO4: Perform Time series Analytics and forecasting

Books/ Journals/ Websites referred:

Pre Lab/ Prior Concepts:

Students should have a basic understanding of: Time series Analytics and forecasting

Procedure:

Data set Used: Hospital_patients_datasets

Step1: Select and Load the dataset

Step2: Convert 'ScheduledDay' and 'AppointmentDay' to datetime format

Step 3: Forecasting Daily Attendance

Step4: Initialize Prophet model for forecasting

Step 5: Fit the model

Step 6: Predict future attendance

Step 7: Plot the forecast

Step 8: Exploratory Data Analysis Functions

Step 9: Running the analysis functions

Implementation details:

Read DataSet

```
[5] df = pd.read_csv('/content/MaunaLoaDailyTemps.csv')
    df.dropna(inplace= True)
    df.reset_index(drop=True, inplace=True)
```

```
[6] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1821 entries, 0 to 1820
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   DATE      1821 non-null   object
 1   MinTemp   1821 non-null   float64
 2   MaxTemp   1821 non-null   float64
 3   AvgTemp   1821 non-null   float64
 4   Sunrise   1821 non-null   int64
 5   Sunset    1821 non-null   int64
dtypes: float64(3), int64(2), object(1)
memory usage: 85.5+ KB
```

```
[7] df.head()
```

	DATE	MinTemp	MaxTemp	AvgTemp	Sunrise	Sunset
0	1/1/2014	33.0	46.0	40.0	657	1756
1	1/2/2014	35.0	50.0	43.0	657	1756
2	1/3/2014	36.0	45.0	41.0	657	1757
3	1/4/2014	32.0	41.0	37.0	658	1757
4	1/5/2014	24.0	38.0	31.0	658	1758

```
df=df[["DATE","AvgTemp"]]
df.head()
```

	DATE	AvgTemp
0	1/1/2014	40.0
1	1/2/2014	43.0
2	1/3/2014	41.0
3	1/4/2014	37.0
4	1/5/2014	31.0

Change Column Names for FB Prophet

```
[9] df.columns = ['ds','y']
```

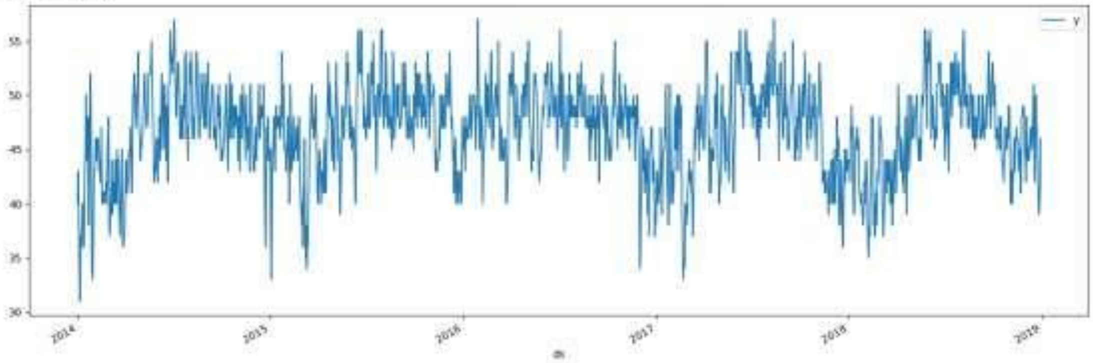
```
df['ds'] = pd.to_datetime(df['ds'])
df.tail()
```

	ds	y
1816	2018-12-26	40.0
1817	2018-12-27	39.0
1818	2018-12-28	40.0
1819	2018-12-29	42.0
1820	2018-12-30	46.0

Plot Your Data

```
df.plot(x='ds',y='y',figsize=(18,6))
```

```
<Axes: xlabel='ds'>
```



Start Making Predictions for next 365 days

Double-click (or enter) to edit

```
m = Prophet()
m.fit(df)
future = m.make_future_dataframe(periods=365) [illegible]
forecast = m.predict(future)
```

```
[illegible]
```

```
forecast.tail()
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_upper	weekly	weekly_lower	weekly_upper	yearly	yearly_lower	yearly_upper	multiplicative_terms
2181	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0
2182	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0
2183	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0
2184	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0
2185	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0

```
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

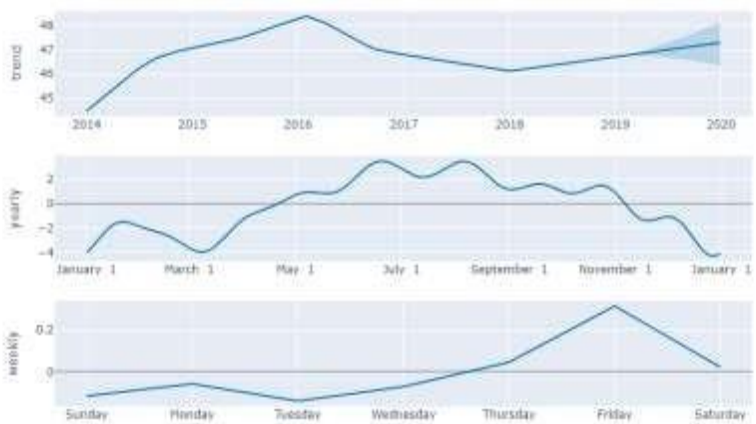
	ds	yhat	yhat_lower	yhat_upper
2181	2019-12-26	43.081677	38.565394	47.496512
2182	2019-12-27	43.317497	39.169191	47.884339
2183	2019-12-28	43.020220	38.625818	47.290154
2184	2019-12-29	42.901558	38.444119	47.290737
2185	2019-12-30	43.004153	38.818901	47.064383

```
df.tail()
```

	ds	y
1816	2018-12-26	40.0
1817	2018-12-27	39.0
1818	2018-12-28	40.0
1819	2018-12-29	42.0
1820	2018-12-30	46.0

USING BUILT-IN FB PROPHET VISUALIZATION plotly

```
# prompt: USING BUILT-IN FB PROPHET VISUALIZATION plotly

plot_plotly(m, forecast)
plot_components_plotly(m, forecast)
```



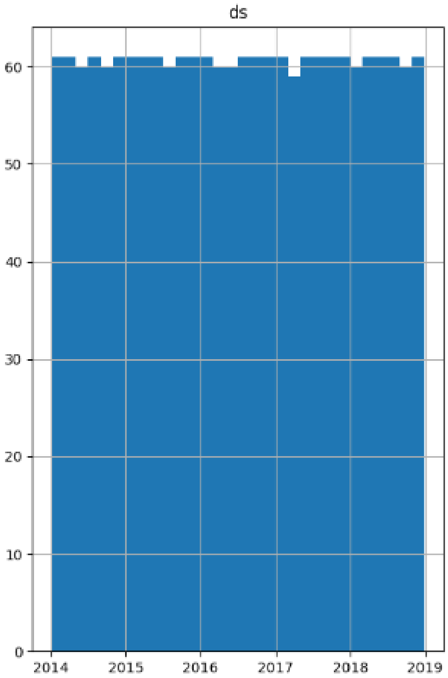
PERFORM EDA ON THE DATASET

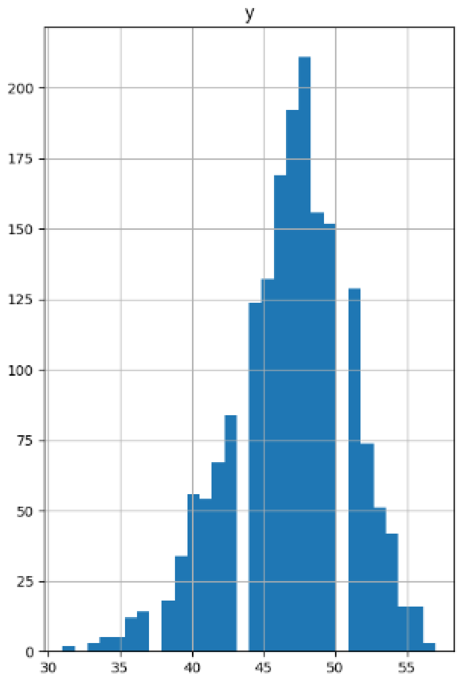
Generate PERFORM EDA ON THE DATASET

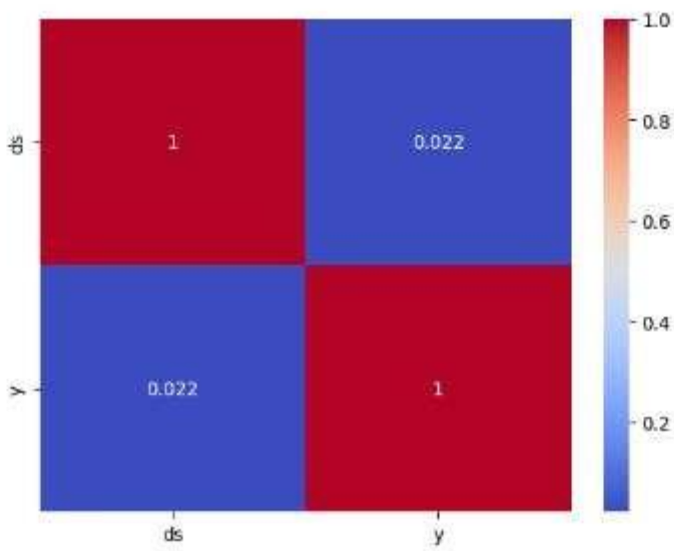
1 of 1 Use code with caution

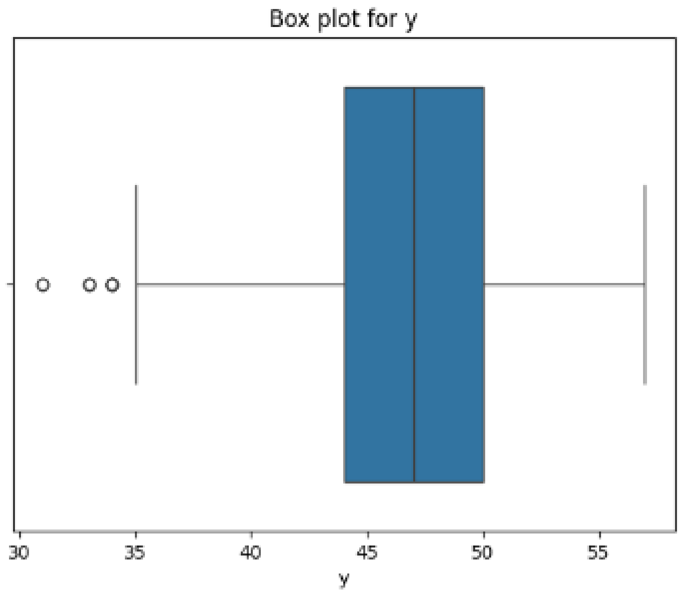
```
# prompt: PERFORM EDA ON THE DATASET

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Basic statistics
print(df.describe())

# Check for missing values
print(df.isnull().sum())

# Explore data distribution
df.hist(bins=30, figsize=(12, 8))
plt.show()

# Correlation matrix (if applicable)
correlation_matrix = df.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.show()

# Box plots to identify outliers
for column in df.columns:
    if pd.api.types.is_numeric_dtype(df[column]):
        plt.figure()
        sns.boxplot(x=df[column])
        plt.title(f'Box plot for {column}')
        plt.show()

# Time series analysis
plt.figure(figsize=(15, 5))
plt.plot(df['ds'], df['y'])
plt.xlabel('Date')
plt.ylabel('Average Temperature')
plt.title('Time Series Plot of Average Temperature')
plt.show()
```

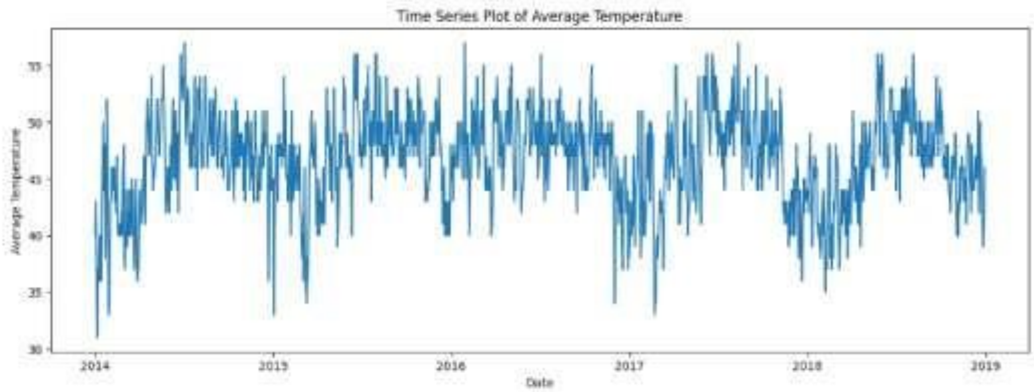
```
                                  ds            y
count                           1821  1821.000000
mean   2016-06-30 20:15:25.205930752    46.818781
min              2014-01-01 00:00:00    31.000000
25%              2015-04-02 00:00:00    44.000000
50%              2016-07-01 00:00:00    47.000000
75%              2017-09-30 00:00:00    50.000000
max              2018-12-30 00:00:00    57.000000
std                              NaN     4.143192
ds    0
y     0
dtype: int64
```


ds
60
50
40
30
20
10
0
2014
2015
2016
2017
2018
2019


y
200
175
150
125
100
75
50
25
0
30
35
40
45
50
55


ds
y
1
0.022
0.022
1
ds
y
1.0
0.8
0.6
0.4
0.2


Box plot for y
30
35
40
45
50
55
y


Time Series Plot of Average Temperature
Average Temperature
30
35
40
45
50
55
2014
2015
2016
2017
2018
2019
Date

Post Lab Descriptive Questions:

1. **Explain the components of time series?**

   A time series typically consists of four main components:

   - **Trend:** This represents the long-term movement in the data. It shows the overall direction (increasing, decreasing, or constant) over time.
   - **Seasonality:** This refers to the regular, periodic fluctuations that occur at specific intervals, such as daily, monthly, or yearly. These patterns are often influenced by external factors, such as holidays or seasons.
   - **Cyclic Patterns:** Unlike seasonality, cyclic patterns occur over irregular intervals and are influenced by economic or other factors. They reflect long-term economic cycles and can last for several years.
   - **Irregular (or Noise):** This component captures random variations or noise in the data that cannot be attributed to trend, seasonality, or cyclic behavior. It often represents unforeseen events or outliers.

2. **How do you handle seasonality in time series data? What methods or transformations can you apply?**

   There are several methods to address seasonality in time series data:

   - **Seasonal Decomposition:** This method involves decomposing the time series into its trend, seasonal, and residual components (e.g., using Seasonal-Trend decomposition using LOESS - STL).
   - **Differencing:** Seasonal differencing can help remove seasonal patterns. For instance, subtracting the value from the same season in the previous year (e.g., yt−yt−sy_t - y_{t-s}yt−yt−s, where sss is the seasonality period).
   - **Fourier Transformations:** These can be applied to model seasonality by capturing periodic fluctuations, useful in more complex seasonal patterns.
   - **Dummy Variables:** Creating dummy variables for seasonal periods can also help incorporate seasonality in regression-based models.
   - **Using Seasonal Models:** Models like SARIMA (Seasonal ARIMA) and Holt-Winters Exponential Smoothing explicitly account for seasonality.

3. What are some common metrics for evaluating forecasting models (e.g., MAE, RMSE, MAPE)?

There are several key metrics used to evaluate the performance of forecasting models:

- **Mean Absolute Error (MAE):** This measures the average magnitude of the errors in a set of forecasts, without considering their direction. It is calculated as:

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t|$$

- **Root Mean Squared Error (RMSE):** This measures the square root of the average of squared differences between predicted and actual values. It gives a higher weight to larger errors:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}$$

- **Mean Absolute Percentage Error (MAPE):** This metric expresses the error as a percentage of the actual values, making it easy to interpret:

$$\text{MAPE} = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right|$$

Each of these metrics has its strengths and weaknesses, and the choice of metric may depend on the specific context of the forecasting task.

Date:______________ **Signature of faculty in-charge**