Importing necessary library

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.statespace.sarimax import SARIMAX
from pmdarima import auto_arima
import warnings
warnings.filterwarnings("ignore")

#Load specific evaluation tools
from sklearn.metrics import mean_squared_error
from statsmodels.tools.eval_measures import rmse
```

Read CSV data file named " **DengueCases.csv** " from local storage and display the primary five rows of the given dataset.

```
disease = pd.read_csv('C:/Users/Ruposh/Desktop/DengueCases.csv',
                       index_col ='Time Period',
                       parse_dates = True)
disease.head()
```

ETS Decomposition and plot

```
result = seasonal_decompose(disease['Cases'], model ='additive')
result.plot()
plt.show()
```

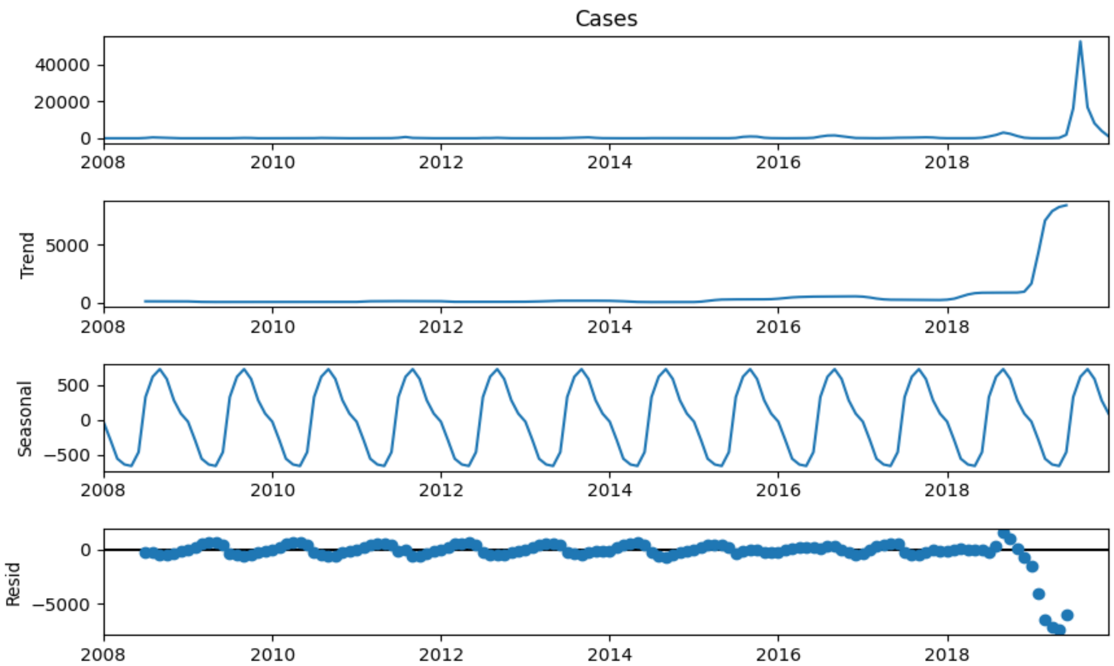

Fig 5.1: ETS Decomposition

Code: Parameter Analysis for the ARIMA model

Fit auto_arima function to **DengueCases** dataset

```
stepwise_fit = auto_arima(disease['Cases'], start_p = 1, start_q = 1,
                          max_p = 3, max_q = 3, m = 12,
                          start_P = 0, seasonal = True,
                          d = None, D = 1, trace = True,
                          error_action ='ignore',
                          suppress_warnings = True,
                          stepwise = True)

#Display summary
stepwise_fit.summary()
```

Output:

```
                                      SARIMAX Results
==========================================================================================
Dep. Variable:                                  y   No. Observations:                  144
Model:             SARIMAX(1, 0, 0)x(0, 1, 0, 12)   Log Likelihood               -1285.509
Date:                            Wed, 30 Sep 2020   AIC                           2575.019
Time:                                    15:06:59   BIC                           2580.784
Sample:                                         0   HQIC                          2577.362
                                            - 144
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.5262      0.024     22.215      0.000       0.480       0.573
sigma2      1.697e+07   4.07e+05     41.744      0.000    1.62e+07    1.78e+07
===================================================================================
Ljung-Box (Q):                        2.06   Jarque-Bera (JB):             43035.13
Prob(Q):                              1.00   Prob(JB):                         0.00
Heteroskedasticity (H):            3583.21   Skew:                             8.51
Prob(H) (two-sided):                  0.00   Kurtosis:                        89.80
===================================================================================
```

Code: Fit ARIMA Model to DengueCases dataset

```
#Split data into train/test set
train = disease.iloc[:len(disease)-12]
test = disease.iloc[len(disease)-12:] # set one year(12 months) for
testing

#Fit a SARIMAX(0, 1, 1)x(2, 1, 1, 12) on the training set
model = SARIMAX(train['Cases'],
                order = (0, 1, 1),
                seasonal_order =(2, 1, 1, 12))

result = model.fit()
result.summary()
```

Output:

```
                                      SARIMAX Results
==========================================================================================
Dep. Variable:                              Cases   No. Observations:                  132
Model:             SARIMAX(0, 1, 1)x(2, 1, 1, 12)   Log Likelihood                -799.988
Date:                            Wed, 30 Sep 2020   AIC                           1609.975
Time:                                    15:46:56   BIC                           1623.871
Sample:                                01-01-2008   HQIC                          1615.618
                                     - 12-01-2018
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1          0.3535      0.047      7.520      0.000       0.261       0.446
ar.S.L12      -0.9627      0.296     -3.247      0.001      -1.544      -0.382
ar.S.L24      -0.3864      0.239     -1.617      0.106      -0.855       0.082
ma.S.L12       0.0635      0.357      0.178      0.859      -0.637       0.764
sigma2      3.738e+04   2062.508     18.121      0.000    3.33e+04    4.14e+04
===================================================================================
Ljung-Box (Q):                       45.67   Jarque-Bera (JB):               453.53
Prob(Q):                              0.25   Prob(JB):                         0.00
Heteroskedasticity (H):               6.84   Skew:                             0.70
Prob(H) (two-sided):                  0.00   Kurtosis:                        12.46
===================================================================================
```

Code: Predictions of ARIMA Model against the test set

```
start = len(train)
end = len(train) + len(test) - 1
predictions = result.predict(start, end,
                             typ = 'levels').rename("Predictions")

#Actual line vs Prediction line
test['Cases'].plot(legend = True , label='Observed')
predictions.plot(legend = True)
plt.title('Evaluate Arima Model (Actual vs Prediction)')
plt.ylabel('Number of Cases')
plt.show()
```

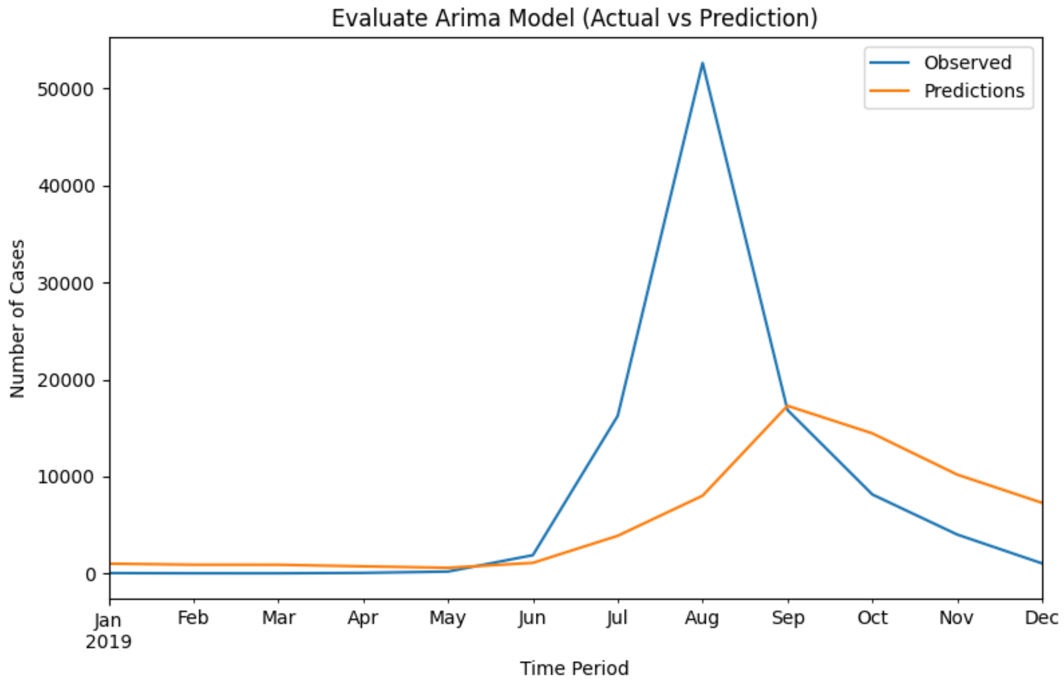

Fig 5.2: Arima Model Evaluation

To understand the accuracy of ARIMA model, we compare predicted cases to actual cases on one-year data. In that case, the prediction accuracy will increase based on the availability of data volume.

Code: Evaluate the model using MSE and RMSE

```
#Calculate root mean squared error (RMSE)
rmse(test["Cases"], predictions)

#Calculate mean squared error (MSE)
mean_squared_error(test["Cases"], predictions)
```

Output:

```
>>> rmse(test["Cases"], predictions)
16430.845732772163
>>> mean_squared_error(test["Cases"], predictions)
269972691.4941572
```

Code: Forecast using ARIMA Model

```
#Train the model on the full dataset
model = model = SARIMAX(disease['Cases'],
                        order = (0, 1, 1),
                        seasonal_order =(2, 1, 1, 12))
result = model.fit()

# Forecast for the next 3 years
forecast = result.predict(start = len(disease),
                          end = (len(disease)-1) + 3 * 12,
                          typ = 'levels').rename('Forecast')

#Forecasting plot
disease['Cases'].plot(figsize = (12, 5), label='Observed', legend = True)
forecast.plot(legend = True)
plt.title('Probable Dengue Outbreaks')
plt.ylabel('Number of Cases')
plt.show()
```

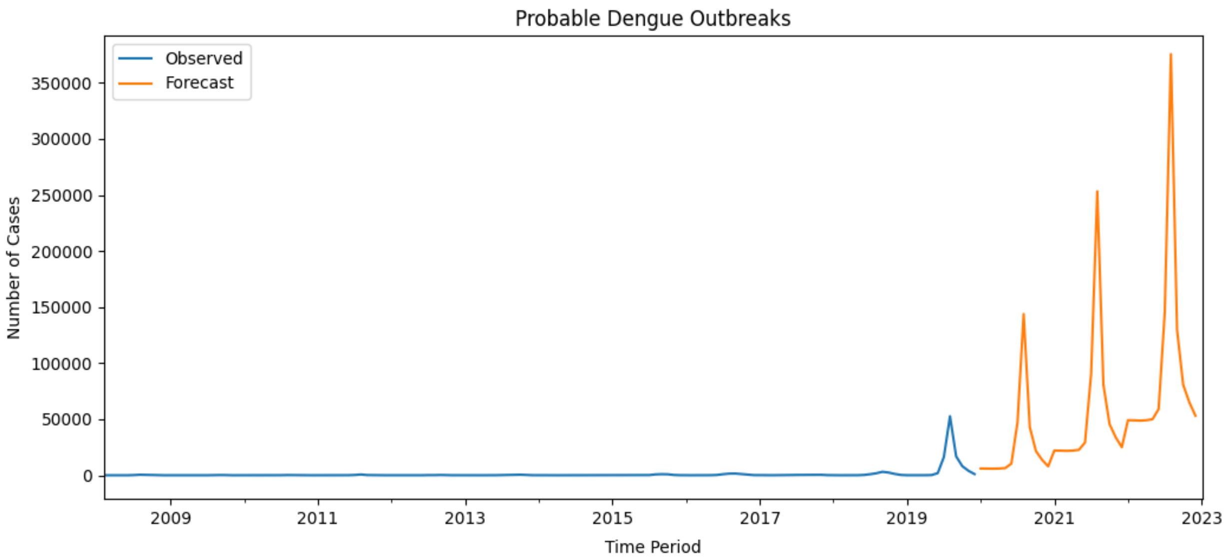

Fig 5.3: Forecasting of probable dengue outbreaks.
