# FLOOD MONITORING SYSTEM

## TEAM MEMBER

**911721104303-S.KARTHIK RAJA**

## Phase-2 Submission Document

**PROJECT:** Flood Monitoring Prediction

## INTRODUCTION

There are few places on Earth where people need not be concerned about flooding. Any place where rain falls is vulnerable, although rain is not the only impetus for flood. A flood occurs when water overflows or inundates land that's normally dry. This can happen in a multitude of ways. Most common is when rivers or streams overflow their banks. Excessive rain, a ruptured dam or levee, rapid ice melting in the mountains, or even an unfortunately placed beaver dam can overwhelm a river and send it spreading over the adjacent land, called a floodplain. Coastal flooding occurs when a large storm or tsunami causes the sea to surge inland.

According to reports from the World Meteorological Organization (2009), approximately 70% of all disasters occurring in the world are related to hydro-meteorological events. Among the disasters, flooding probably is one of the most severe disasters affecting the people across the globe.

India is the worst flood affected country in the world after Bangladesh and accounts for one- fifth of global death count due to floods. Nearly 75 percent of the total Indian rainfall is concentrated over a short monsoon season of four months (June-September).As a result, the rivers witness a heavy discharge during these months, leading to widespread floods. About 40 million hectares of land in the country is liable to floods according to National Flood Commission, and an average of 18.6 million hectares of land is affected annually.

# Geospatial dataset to make flood research easier

**by Sahana Ghosh on 16 November 2021**

- India's first digital flood inventory has digitised flooding events from 1985 to 2016 using data from the India Meteorological Department.
- The digital inventory will aid future data collection efforts, scientific research on floods, disaster risk reduction, loss and damage estimates and pave the way to develop financial products to tackle disaster risks.
- Stakeholders working to build financial products and mechanisms such as disaster risk insurances including crop insurances could also benefit from the dataset.

Data on fatalities and damage associated with India's flooding events from 1985 to 2016 have been meticulously mined from printed government records and combined with information from global databases to create India's first digital flood inventory freely available in modern geospatial formats.

The India Flood Inventory (IFI) pools 89% of its data from the India Meteorological Department's Disastrous Weather Events, and the remaining from Emergency Events Database (EM-DAT) and Dartmouth Flood Observatory (DFO), states the paper documenting the inventory's inception.

"We have taken decades of data from the published document by IMD's DWE, digitised it, fixed a lot of things, and augmented it with geospatial information in the form of GIS files. Each historical flooding event now has its information digitised, which IMD didn't have till now," said Manabendra Saharia from IIT-Delhi's Hydrosense Lab, who led the development of the inventory.

The objective was to provide a ready-made database to the community that could be used for future work including disaster management personnel and scientists working on flood research and management. The inventory will also aid future data collection efforts.

## FLOOD EARLY WARNING SYSTEM:

As the name indicates, Flood Early Warning System (FLEWS) is a system by which flood induced hazards can be minimized and prevented. Currently different organizations are working on flood forecasting and early warning at national, continental and global scale.

In a flood early warning system the most important input is real time hydro-meteorological observations provided by weather radar satellites and automatic hydro-meteorological station network (Billa et al.,2006; Budhakooncharoen, 2004). This real time data can be used in various ways to evaluate flood risks and issues of flood warning. Apart from real time data, probabilisticweather forecasts (Numerical Weather Prediction-NWP) are also playing an important role in providing input for hydrological models to generate warnings scenarios (Burger et al.2009; Thielen et al., 2010). Besides having forecasts of the most important input (precipitation), a model needs to be selected that characterizes and simulates the catchment responses for flood early warning.

## BENEFITS OF FLOOD EARLY WARNING SYSTEMS:

The development of flood forecasting and warning systems is an essential element in regional and national flood preparedness strategies, and is a high priority in many countries. Flood EWS are being considered as an alternative for dealing with flood problems, partly because these systems are less expensive compared to structural schemes. Despite the high priority accorded to flood warnings in flood risk management by governments, there is a lack of good data on thebenefits and costs of these systems (Wallingford 2006).

The benefits of an early warning system can be calculated by assessing the possible savings of the quantity of flood damage to private and public assets resulting from action taken in response to the warning. This flood reduction benefit BEWS can be expressed as:

$$BEWS=Xwithout-Xwith$$

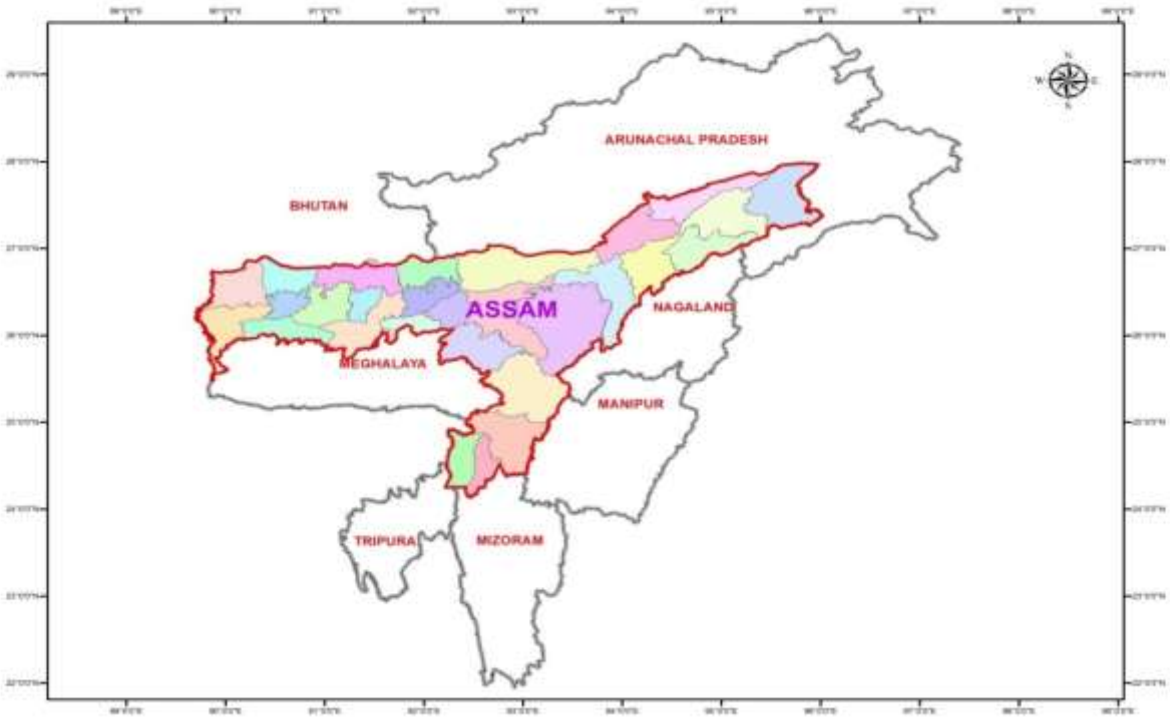
Where, Xwithout=without project economic flood damage; and

Xwith= economic damage if the project is implemented

The benefits of flood early warning systems come from the savings in flood damages. Floods arerandom events that cause damages and hence flood damages are also random or probabilistic events: the probability of any specific amount of flood damage depends on the probability of the flood event necessary to cause those damages. Determining flood damages combines a risk assessment in terms of the probability of future flood events to be averted, and a vulnerability assessment in terms of the damage that would be caused by those floods and, therefore, the economic savings to be gained by their reduction.

## ABOUT ASSAM:

Assam, the gateway to the northeastern part of India, extending from $22^0 19'$ to $28^0 16'$ North Latitude and $89^0 42'$ to $96^0 30'$ East Longitude is situated between the foot hills of the Eastern Himalayas and the Patkai and Naga ranges. Assam is bordered in the North and East by Bhutan and Arunachal Pradesh. Along the south lie Nagaland, Manipur and Mizoram. Occurrence of flood has been an age-old phenomenon in the riverine areas of this region.



The frequency and intensity of floods has grown over the years primarily because of the increased encroachment of flood plains. Interestingly, while the number of deaths caused by flooding has decreased over the last decade, the number of affected populations and economic losses has not decreased significantly. While the State has come forward to take up mitigation measures like construction of raised platforms, embankments etc, floods still continue to be a menace in the State causing colossal damage to property and disrupting the livelihoods of the poor people in the State year after year. The rural population thus suffers from loss of crops and property due to annual flood, which affect their income and further increases their vulnerability to endemic poverty.

Assam, is in fact one of the poorest State with approximately 36% of the population living below poverty line. The state also lags behind in many other development indicators. Several factors are responsible, including poor infrastructure, remoteness, and inability to minimize the impacts of damages and loss of productivity from frequent flooding & other natural calamities.

## STATEMENT OF PROBLEM:

Flood forecasting and early warning is used for alerting the likely damage center well in advance of the actual arrival of flood, to

enable the people to move and also to remove the movable properties to safer places or to raised platforms specially constructed for this purpose. Flood is an annual event in the State of Assam. More than 40 percent of its land surface is susceptible to flood damage. The total flood-prone area in the Brahmaputra valley is about 3.2 Mha. (Goswami, 2001). The Brahmaputra valley had experienced major floods in 1954, 1962, 1966, 1972, 1974, 1978, 1983, 1986, 1988, 1996, 1998, 2000, 2004 and 2007 & 2012 which clearly shows that floods are an annual event in the State. This affects a large section
of the people of the riverine areas leaving them to cope with their annual losses.

Assam, inspite of suffering from annual flood events, unfortunately did not have a system of any early warning mechanisms that would alert the concerned districts/circles/villages from the occurrence of a disaster. The existing disaster management mechanism is primarily focused on strengthening rescue and relief arrangements during and after major flood disasters. Little work has been done in a scientific context on minimizing the incidence and extent of flood damage.
To minimize flood damage the basic approach is to prevent floodwaters from reaching the vulnerable centres. The Central Water Commission (CWC) under the Ministry of Water Resources issues flood forecasts and warnings.

However, CWC gives the water level of only the major rivers of the State which does not indicate the areas/villages where the flooding would occur and leaves the administrative machinery clueless as to which village or revenue circle should be warned /evacuated. The government felt an inadequacy in the early warning system and therefore thought for the development of a flood early warning system and/or decision tools which relies on hydrological modeling and the use of near real time data and consulted different stakeholders to find a solution to the problem.

## EXPLAINATION PROGRAM:

Creating a Flood Monitoring System with predictive modeling and historical flood data to improve early warnings is a complex task that requires access to historical data, machine learning models, and real-time data acquisition systems. Below, I'll provide you with a high-level outline of how such a system could be designed, along with a Python code snippet demonstrating the predictive modeling aspect using a simple linear regression model as an example.

1. **Data Collection**:

   - Acquire historical flood data, including information on water levels, rainfall, river discharge, weather conditions, and past flood events.

   - Set up real-time data acquisition systems (sensors, weather APIs, river gauges, etc.) to continuously collect current data.

2. **Data Preprocessing**:

   - Clean and preprocess historical data, handling missing values and outliers.

   - Organize the data into a format suitable for modeling.

3. **Feature Engineering**:

   - Create relevant features from historical and real-time data. For example, you could calculate rolling averages of rainfall, river discharge trends, etc.

4. **Machine Learning Model**:

   - Train a predictive model (e.g., a regression model, time series model, or deep learning model) using historical data.

   - Use this model to predict future water levels based on real-time data and historical patterns.

Here's a simplified example of how to implement a predictive model using linear regression with historical flood data in Python:

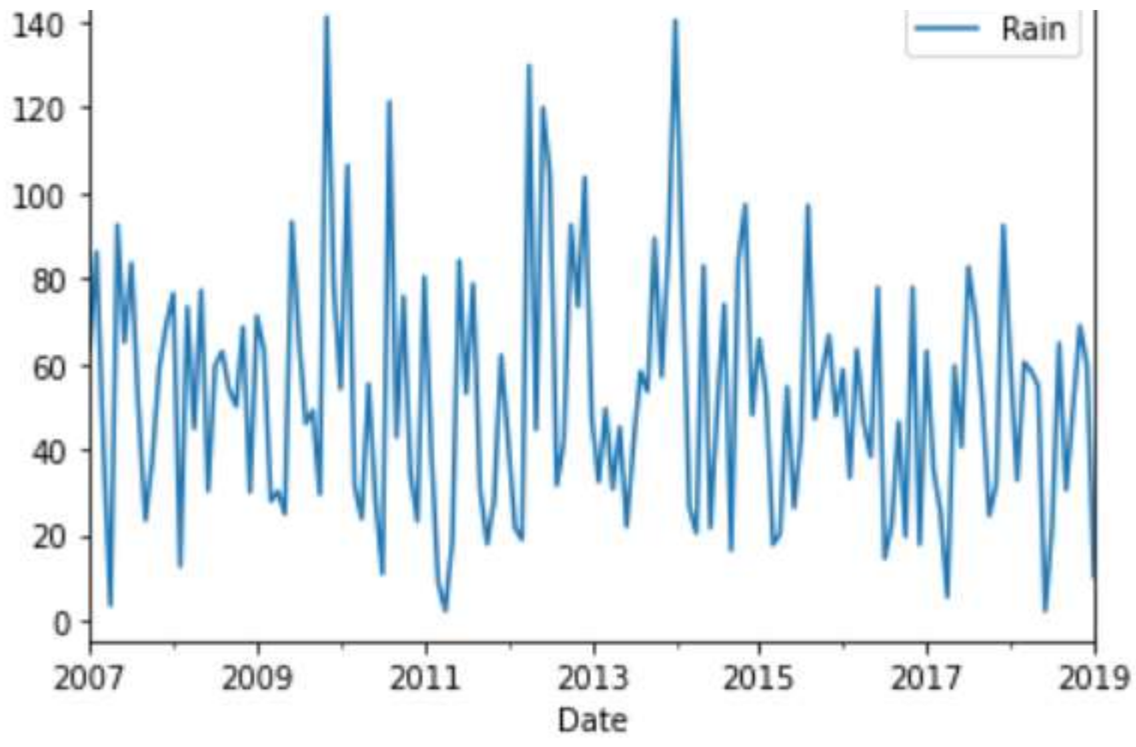
~~~
    ```python
    import pandas as pd
    from sklearn.model_selection import train_test_split
    from sklearn.linear_model import LinearRegression
    from sklearn.metrics import mean_squared_error

# Load historical flood data (assuming you have a CSV file with relevant features)
data = pd.read_csv('historical_flood_data.csv')


# Split the data into training and testing sets
    X = data[['Rainfall', 'RiverDischarge', 'WeatherCondition']]  # Features
    y = data['WaterLevel']  # Target variable
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42)

    # Train a linear regression model
    model = LinearRegression()
    model.fit(X_train, y_train)

    # Make predictions on the test set
    y_pred = model.predict(X_test)

    # Evaluate the model's performance
    mse = mean_squared_error(y_test, y_pred)
    print(f"Mean Squared Error: {mse}")

    # Predict water levels for a new observation (real-time data)
    new_data = pd.DataFrame({'Rainfall': [10.2], 'RiverDischarge': [150.5],
~~~

```
'WeatherCondition': [1]})
predicted_water_level = model.predict(new_data)
print(f"Predicted Water Level: {predicted_water_level[0]}")
```

In this example, we load historical flood data, split it into training and testing sets, train a linear regression model, and use it to make predictions. In a real-world scenario, you would continuously collect real-time data and use the trained model to make predictions for early flood warnings.

Remember that building a robust Flood Monitoring System with predictive modeling requires extensive data, expertise in machine learning, and integration with real-time data sources and alerting mechanisms.

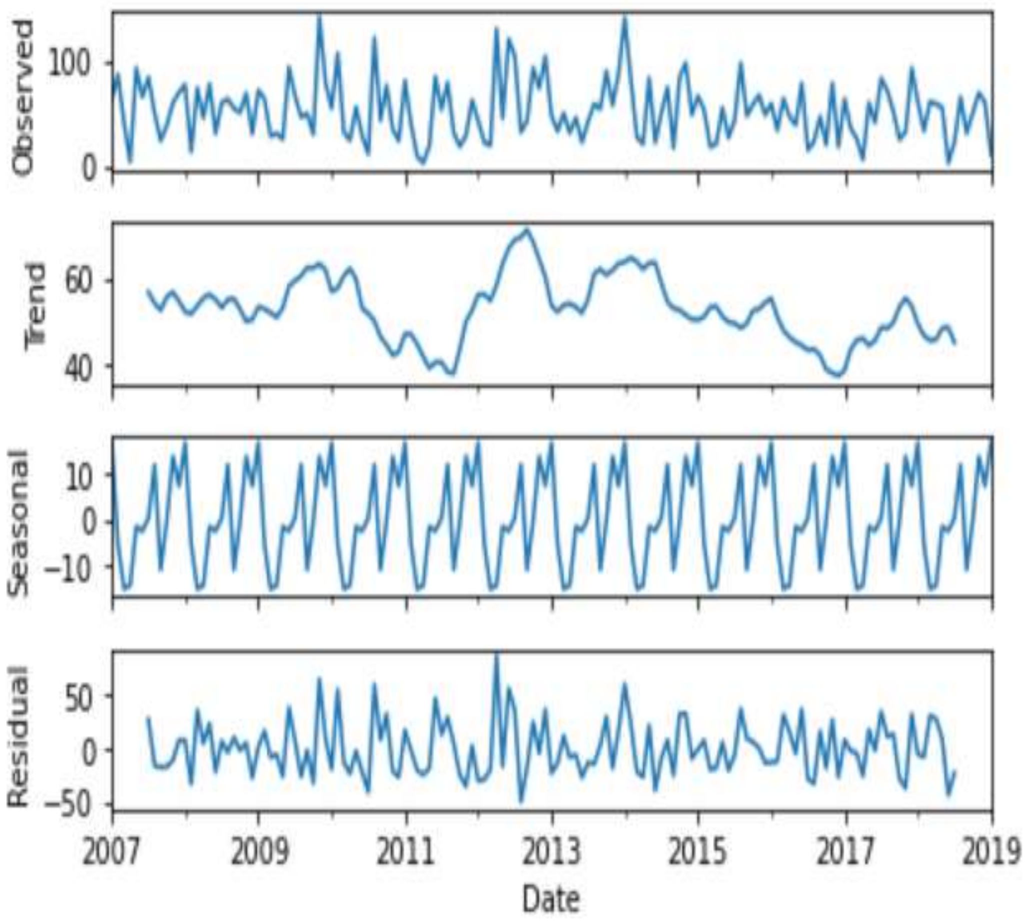
## ANALYSING THE DATA :



One of the major issues that popped up was the data type of the date column. After tonnes of digging around in stack overflow, I found the solution was to convert it to a timestamp then converted back into a Date Time format. I think this has to do with the changed data frame into a monthly data frame so it must have messed up the data type which is why I had to change it again.

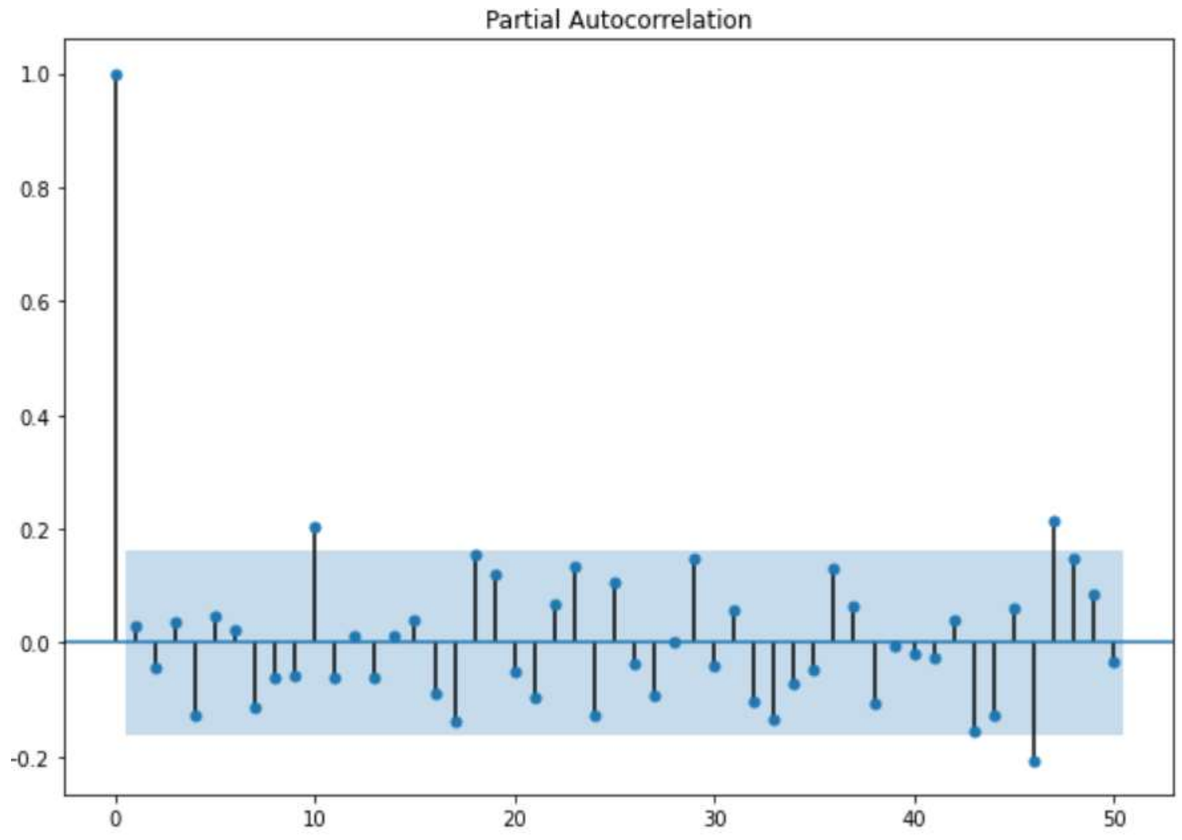
A minor thing I had to adjust was the index because when I first plotted the graphs the forecast did not provide the date only providing an increasing numerical number. So, I went to the tutorial's notebook and her dataframe had the date as the index. So, I changed my dataset, so the index contains the dates so when the forecast is plotted the dates are shown on the x-axis.

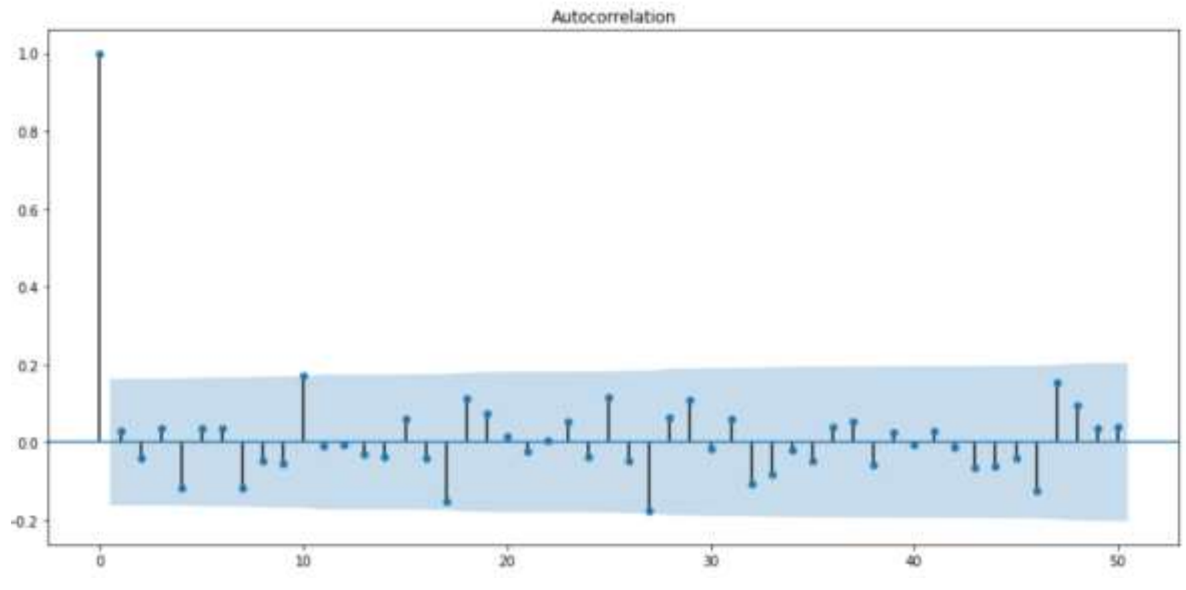
Now for the analysis. This is a time-series analysis as we are doing forecasting. I found this article here which I followed. I used the statsmodels package. Which helps provide models for statistical analysis. First, we did a decomposition which separated the dataframe into a trend, seasonal and residual components.



Next, the tutorial asks us to check if the time series is stationary. In the article, it's defined as "A time series is stationary when its statistical properties such as mean, variance, and autocorrelation are constant over time. In other words, the time series is stationary when it is not dependent on time and not have a trend or seasonal effects."

To check if the data is stationary, we used autocorrelation function and partial autocorrelation function plots.





There is a quick cut off the data is stationary. The Autocorrelation and

Partial autocorrelation functions give information about the reliance of time series values.

Now we used another python package called pmdarima. Which will help me decide my model.

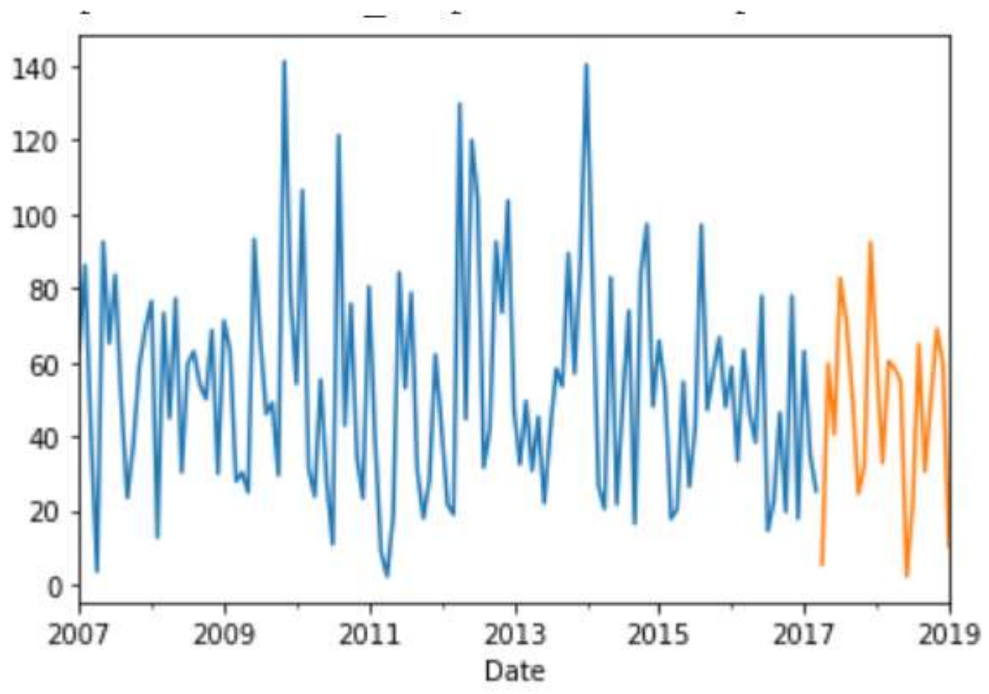
```
import pmdarima as pm


model = pm.auto_arima(new_index_df_new_index['Rain'], d=1, D=1,
                m=12, trend='c', seasonal=True,
                start_p=0, start_q=0, max_order=6, test='adf',
                stepwise=True, trace=True)
```

All of the settings were taken from the tutorial. I will let the tutorial explain the numbers:

"Inside auto_arima function, we will specify d=1 and D=1 as we differentiate once for the trend and once for seasonality, m=12 because we have monthly data, and trend='C' to include constant and seasonal=True to fit a seasonal-ARIMA. Besides, we specify trace=True to print status on the fits. This helps us to determine the best parameters by comparing the AIC scores."

After than I spilt the data into train and test batches.

```
train_x = new_index_df_new_index[:int(0.85*(len(new_index_df_new_
index)))]

test_x = new_index_df_new_index[int(0.85*(len(new_index_df_new_i
ndex))):]
```



When Splitting the data for the first time I used SciKit Learn’s train_test_split function to split the data. But this led to some major errors later on when plotting the data so I'm using the tutorial method.

Then we trained a SARIMAX based on the parameters produced from earlier.

```
from statsmodels.tsa.statespace.sarimax import SARIMAX


model = SARIMAX(train_x['Rain'],
            order=(2,1,0),seasonal_order=(2,1,0,12))
results = model.fit()
results.summary()
```

## PLOTTING THE FORECAST :

Now we can start work on forecasting as we now have a trained model.

```
forecast_object = results.get_forecast(steps=len(test_x))
mean = forecast_object.predicted_mean
conf_int = forecast_object.conf_int()
```

dates = mean.index

These variables used to help us plot the forecast. The forecast is as long as the test dataset. The mean is the average prediction. The confidence interval gives us a range where the numbers lie. And dates provide an index so we can plot the date.

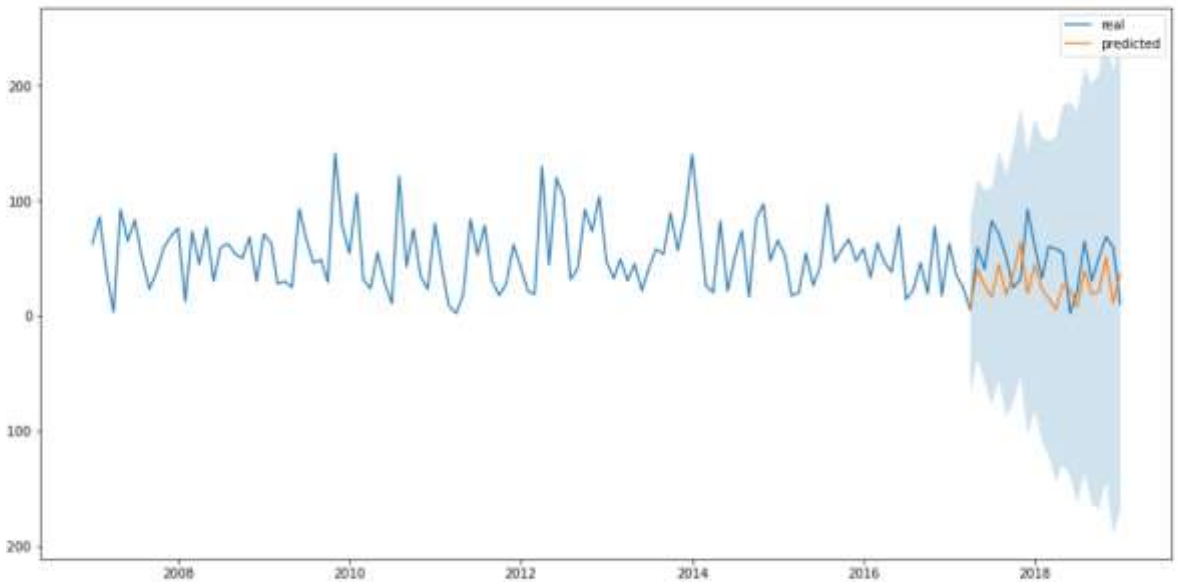
plt.figure(figsize=(16,8))

df = new_index_df_new_index plt.plot(df.index, df, label='real')

plt.plot(dates, mean, label='predicted')

plt.fill_between(dates, conf_int.iloc[:,0], conf_int.iloc[:,1],alpha=0.2)

plt.legend() plt.show()



This is example of an in-sample forecast. Now lets see how we make a out-sample forecast.

pred_f = results.get_forecast(steps=60)

pred_ci = pred_f.conf_int()

ax = df.plot(label='Rain', figsize=(14, 7))

pred_f.predicted_mean.plot(ax=ax, label='Forecast')

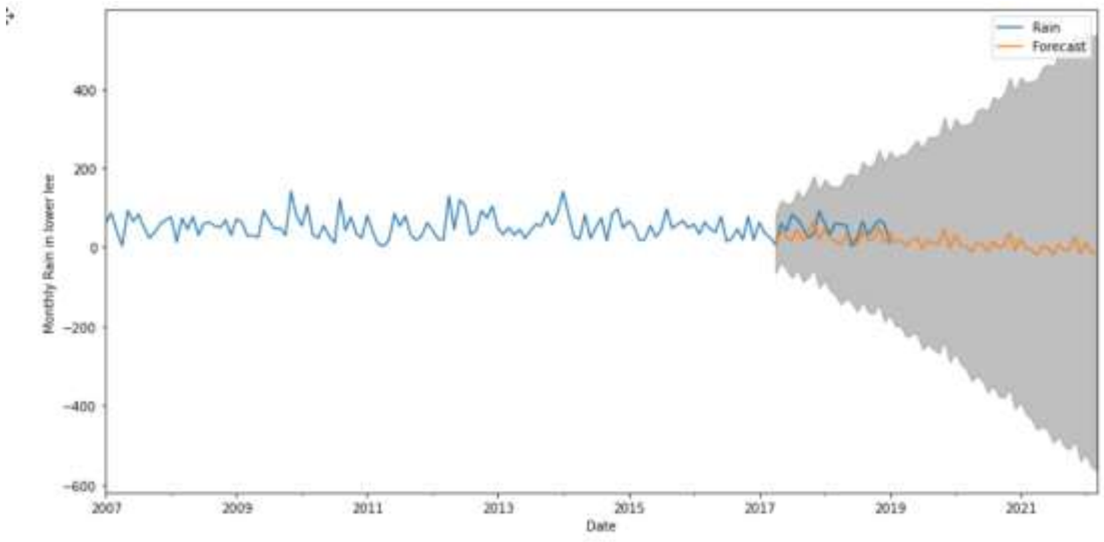
ax.fill_between(pred_ci.index,

    pred_ci.iloc[:, 0],

    pred_ci.iloc[:, 1], color='k', alpha=.25)

ax.set_xlabel('Date')

```
ax.set_ylabel('Monthly Rain in lower lee')
plt.legend()
plt.show()
```



This is forecasting 60 months into the future.

Now we have forecasting data. I needed to work on which area can get flooded.

## GETTING ELEVATION DATA:

To work out areas that are at risk of flooding I had to find elevation data. After googling around. I found that the UK government provide elevation data of the country. Using LIDAR. While I was able to download the data. I worked out that I did not have a way to view the data in python. And I may have to pay and learn a new program called ArcGIS. Which is something I did not want to do.

So I found a simpler alternative using Google Maps API elevation data. Where you can get elevation data of an area. Using coordinates. I was able to access the elevation data using the Python package requests.

```
import requests
r = requests.get('https://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536,-104.9847034&key={}'.format(key))
```

```
r.json()


{'results': [{'elevation': 1608.637939453125,
   'location': {'lat': 39.7391536, 'lng': -104.9847034},
   'resolution': 4.771975994110107}],
 'status': 'OK'}
```

Now we need to work out when the point will get flooded. So using the rainfall data we compare the difference between elevation and rainfall. And if the rain passes elevation then the place is underwater.

```
import json
r = requests.get('https://maps.googleapis.com/maps/api/elevation/json?locations=51.528771,0.155324&key={}'.format(key))
r.json()
json_data = r.json()
print(json_data['results'])
elevation = json_data['results'][0]['elevation']
print('elevation: ', elevation )


rainfall_dates = []
for index, values in mean.iteritems():
    print(index)
    rainfall_dates.append(index)


print(rainfall_dates)
for i in mean:
  # print('Date: ', dates_rain)
  print('Predicted Rainfall:', i)
  print('Rainfall vs elevation:', elevation - i)
  print('\n')
```

```
Predicted Rainfall: 8.427437412467206
Rainfall vs elevation: -5.012201654639448



Predicted Rainfall: 40.91480530998025
Rainfall vs elevation: -37.499569552152494



Predicted Rainfall: 26.277342698245548
Rainfall vs elevation: -22.86210694041779



Predicted Rainfall: 16.720892909866357
Rainfall vs elevation: -13.305657152038599
```

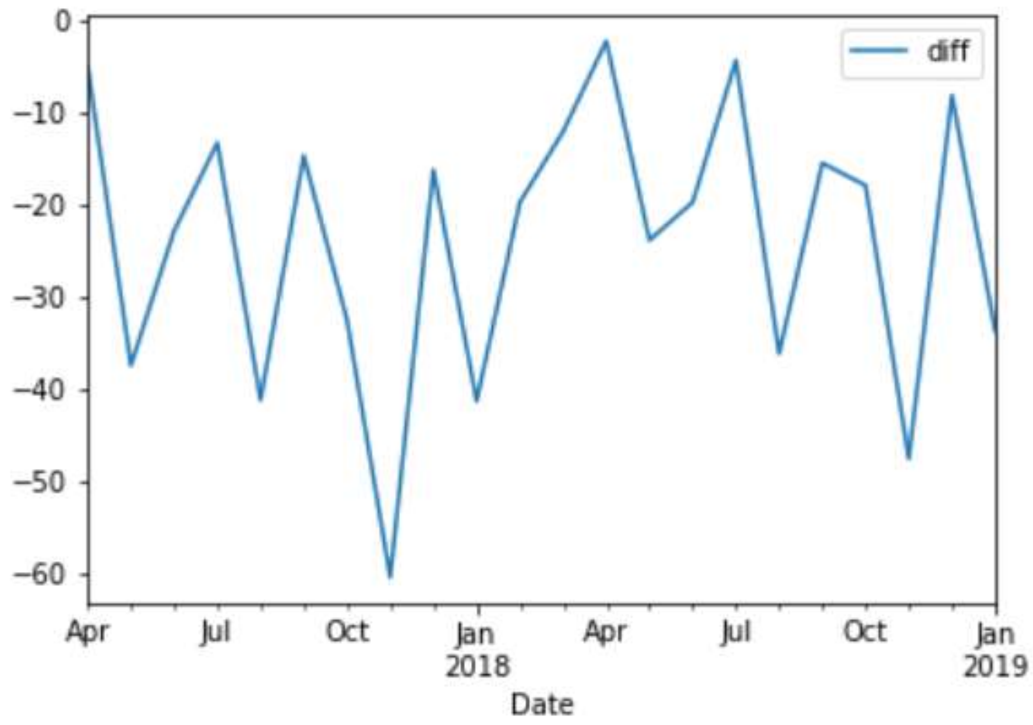
As we can see if the monthly rainfall drops all in one day. Then the area will get flooded.

```
diff_rain_ls = []
for f, b in zip(rainfall_dates, mean):
    print('Date:', f)
    print('Predicted Rainfall:', b)
    diff_rain = elevation - b
    diff_rain_ls.append(diff_rain)
    print('Rainfall vs elevation:', elevation - b)
    print('\n')
    # print(f, b)
```

This allows me to compare the dates with rainfall vs elevation difference.

```
df = pd.DataFrame(list(zip(rainfall_dates, diff_rain_ls)),
              columns =['Date', 'diff'])
df.plot(kind='line',x='Date',y='diff')
```

```
plt.show()
```



I did the same thing with the 60-month forecast

```
rainfall_dates_60 = []
for index, values in mean_60.iteritems():
   print(index)
   rainfall_dates_60.append(index)


diff_rain_ls_60 = []
for f, b in zip(rainfall_dates_60, mean_60):
   print('Date:', f)
   print('Predicted Rainfall:', b)
   diff_rain_60 = elevation - b
   diff_rain_ls_60.append(diff_rain_60)
   print('Rainfall vs elevation:', elevation - b)
   print('\n')
```

## CONCLUSION:

The district administration for those districts under FLEWS has welcomed the early warning and participated in it with great commitment. Flood is an annual event for the State of Assam and therefore the FLEWS help the administration in not only giving early warning but also in reducing the losses. Sustainability of the project is ensured as the project is being expanded to cover the entire State and also because it has proved to be an effective tool for decision making.

The model developed by North East Space Application Centre for Flood Early Warning which has an accuracy rate of around 60% needs to be developed further for which infrastructure for IMD & CWC in terms of increase in rain gauge stations specially in the upper reaches and increase in river gauging stations is required to be undertaken by Government of India. Further the Flood Early Warning System needs to be extended to cover all the flood hazard districts of Assam which will allow individuals exposed to hazard or agencies involved in the management of disaster to take action to avoid or reduce their risk for effective response.