Predicting Dengue Cases in Manila, Philippines

Dengue Fever is an illness that primarily appears in tropical areas. It can be life threatening if not treated. Possible symptoms include flu symptoms, bleeding, and drop in blood pressure.

Over a million cases of dengue infection happen yearly in the world. Using a dataset on dengue in Manila, Phillippines, we will create a model to predict outbreaks of dengue with the weather. Mosquito populations are dependent on the weather, such as temperature, humidity, and precipitation. Our model creates these to predict dengue outbreaks with a fair amount of accuracy.

Ending the epidemics of tropical diseases is part of UN Sustainable Development Goal 3.3. This goal focuses on ending epidemics like AIDS, tuberculousis, and communicable diseases. Our model could be used by local governments and relief groups to help plan disease response teams. Predicting the location of disease outbreaks are vital to providing aid in a timely matter.

Sources: Mayo Clinic

In [1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
```

Importing Data

The Manila Dengue Case Data was obtained from an official Philippines Government Source

The weather data was obtained from www.visualcrossing.com

We did some manipulation in Excel to format the CSV files. The Manila Dengue Case data does not include a row entry when there are no reported cases for a given day. We will do some cleaning to add these dates for our analysis.

In [2]:
```
# Read the file with weather data
weather_df = pd.read_csv("manila_weather.csv")
```

```
# Load Manila dengue cases file in
manila_df = pd.read_csv("Dengue_Manila_2_Columns.csv")
```

In [3]:
```
# Cleaning Manila Dengue Case File

# Extract neccessary columns and Rename
manila_df = manila_df[["MM/DD/YYYY", "Cases"]]
manila_df.rename(columns={"MM/DD/YYYY": "Date"}, inplace=True)
# Convert string Date into DateTime object
manila_df["Date"] = pd.to_datetime(manila_df["Date"], format='%Y-%m-%d')

# For any date with zero cases in manila dataset, the date is not listed
# This section adds in the missing dates with Cases = 0

# First, we will create another dataframe with every date within the range o

# Create list of DateTimes with every date in the range of our data
all_dates = pd.date_range(start="01-01-2018", end="12-2-2023")
# Convert to dataframe
full_dates_df = pd.DataFrame(data = {"Date": all_dates})

# Merge the Data Frames together
# Any empty dates are left with a null value
merged_df = full_dates_df.merge(manila_df, on="Date", how = "left")
# Replace null Cases with the value 0
merged_df["Cases"] = merged_df["Cases"].fillna(0)
# Type cast Cases to int, since they are discrete and were changed implicitl
merged_df["Cases"] = merged_df["Cases"].astype(int)
# Dataset is clean, now we can save it for future use
#clean_manila = merged_df
#clean_manila.to_csv('clean_manila.csv', index=False)

case_df = merged_df
```

In [4]:
```
# Rename date column in weather_df
weather_df.rename(columns = {"datetime":"Date"}, inplace=True)
# Convert string Date into DateTime object
weather_df["Date"] = pd.to_datetime(weather_df["Date"], format='%Y-%m-%d')
```

In [5]:
```
# Converting daily data to weekly data

# These are a aggregation functions
# For temperature and humidity, we will take the average for the week
# For cases and precipitation, we will take the sum for the week
weather_aggregation_functions = {
    'temp': "mean",
    'humidity': "mean",
    "precip": 'sum'
}

case_aggregation_functions = {
    "Cases": "sum",
}

weather_df_copy = weather_df.copy()
```

```
# Aggregate data from days to weeks

# Set index to our Date so we can use aggregate function
# Aggregate Weather data
weather_df_copy.set_index('Date', inplace = True)
weekly_data = weather_df_copy.resample('W').agg(weather_aggregation_functior
weekly_data = weekly_data.reset_index()
weather_df = weekly_data.copy()

case_df_copy = case_df.copy()
# Aggregate Case data
case_df_copy.set_index('Date', inplace = True)
weekly_data = case_df_copy.resample('W').agg(case_aggregation_functions)
weekly_data = weekly_data.reset_index()
case_df = weekly_data.copy()
```

In [6]:
```
# Join the Case data and Weather datasets together on the Week
df = pd.merge(weather_df, case_df, on="Date", how="inner")

# Now, we will create a column for next week's cases
# This is necessary as dengue has an incubation time of 5-7 days
# This means are cases can't be measured until almost a week after a patient
df['Next_week_cases'] = df['Cases'].shift(-1)
df.head()
```

Out[6]:

	Date	temp	humidity	precip	Cases	Next_week_cases
0	2018-01-07	27.385714	73.757143	0.295	60	72.0
1	2018-01-14	27.328571	70.571429	1.263	72	84.0
2	2018-01-21	26.957143	78.114286	10.137	84	62.0
3	2018-01-28	27.000000	82.957143	9.074	62	82.0
4	2018-02-04	26.871429	76.614286	0.341	82	63.0

Predicting Outbreaks

We classify a Dengue outbreak as there being more than the average amount of cases. This ends up being 48 or more cases in Manila.

In [7]:
```
# Here we create the outbreak threshold
# The
outbreak_threshold = df.describe()['Next_week_cases']['mean']
df['outbreak'] = df['Next_week_cases'] > outbreak_threshold
print("Outbreak Threshold (Weekly Cases): " + str(outbreak_threshold))
print("Number of Outbreaks: " + str(len(df[df['outbreak']==True])))
print("Total Weeks: " + str(len(df)))
```

```
Outbreak Threshold (Weekly Cases): 47.32142857142857
Number of Outbreaks: 110
Total Weeks: 309
```

The Model

Independent variables.

Dengue is primarily spread through mosquito bites. Our weather data includes factors that affect the mosquito population.

1. Humidity: High humidity levels are known to facilitate mosquito breeding by maintaining suitable moisture levels for egg development.
2. Precipitation: Rainfall creates breeding sites by filling containers and other water-holding structures where mosquitoes lay their eggs.
3. Temperature: Temperature influences the development rate of mosquito eggs, larvae, and pupae. Higher temperatures generally accelerate their growth and shorten the time it takes for mosquitoes to mature and reproduce.

General Explanation

Our model used the above weather from the week before. It is important to use weather data from the week before because the dengue virus has an incubation time. If we used the weather of the week, the data would be outdated as the cases wouldn't be recorded until the next week.

Another feature used is the cases of the week before. If the Dengue virus is active the week before, it is more likely to spread.

The model used is a Random Forest. Random Forests create many decision trees that create classifications of a variable, in this case whether or not there is an outbreak. These trees are combined into a big one that becomes the model. This is a powerful machine learning algorithm.

In [8]:
```
# https://towardsdatascience.com/random-forest-in-python-24d0893d51c0
rf = RandomForestClassifier(n_estimators = 1000, random_state = 432)

X = df[["temp", "humidity", "precip", "Cases"]]
y = df['outbreak']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rar

# Train the model on the training data
rf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = rf.predict(X_test)

# https://www.w3schools.com/python/python_ml_confusion_matrix.asp
conf_matrix = confusion_matrix(y_test, y_pred)
print(conf_matrix)
```

```
Accuracy = metrics.accuracy_score(y_test, y_pred)
print("Accuracy: "+ str(Accuracy))
```

```
[[36  2]
 [ 3 21]]
Accuracy: 0.9193548387096774
```

Evaluating the Model

The model can be judged by its Confusion Matrix. This matrix shows the 21 True Positives, 36 True Negatives, 2 False Positives, and 3 False Negatives. This gives us just under 92% accuracy, which means the model is effective at predicting outbreaks.

When creating the model, we split the data into training set and a testing set. The model is trained on the training set. The testing set is used to test the model on unseen data, which allows us to judge the model on how it deals with new data. This also allows us to test for overfitting, which is when the model works very well on the training data, but not any other data. Our model generalizes well to new data.

With this model, future weather forecasets could be used to predict upcoming dengue outbreaks in Manila. Our outbreak prediction model should only be used in Manila, but similar models can be created for other cities and regions. These would help local governments respond and prepare for dengue cases and help end the dengue virus. This would help fulfill UN Sustainable Development Goal 3.3.