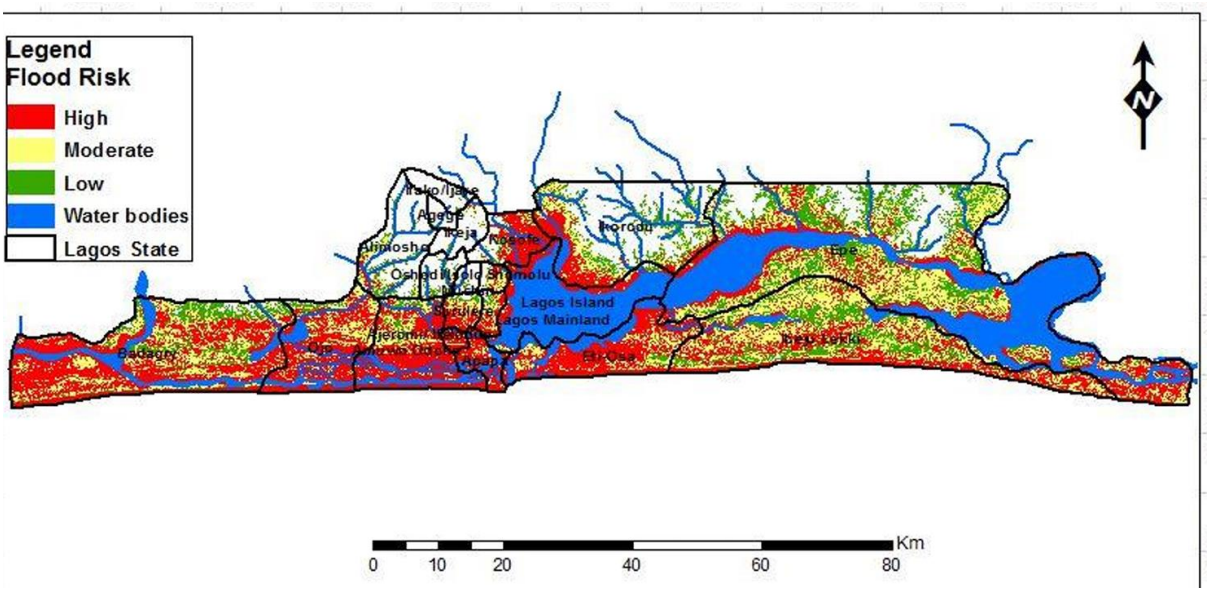
Flood Prediction Analysis for Lagos, Nigeria

Barakat Akinsiku

Introduction

Lagos, Nigeria, a thriving metropolis with over 21 million residents faces a significant annual challenge: flooding. The state lies along the coast of the Atlantic Ocean and its altitude ranges from 77metres above sea level to 25metres below the sea level. The highest altitude in the state is located in Alimosho LGA while the lowest altitude is located in Lagos Island. Other high altitude areas in the state include Ifako/Ijaiye (76m) and Ikorodu (70m) while Ibeju Lekki(-19) and Eti-Osa(-13) are low altitude areas.



During the rainy season, heavy downpours overwhelm drainage systems, transforming streets into waterways and homes into temporary lagoons. This not only disrupts daily life and commerce but can also cause significant damage to infrastructure and property while posing health risks.

Understanding and predicting flood events are crucial for mitigating their impact. This report aims to develop a flood prediction model for Lagos by analyzing historical data on various factors that contribute to flooding. The model aims to provide accurate predictions that can be used to enhance flood risk management strategies, allowing authorities to take timely measures to reduce the impact of flooding.

Data Acquisition and Analysis

Historical Data

The analysis utilizes historical weather data which includes variables like precipitation, wind speed, humidity, and moon phase. The primary data sources were government agencies and reputable weather data providers such as Visual Crossing from which the weather data of 2002 to 2024 was obtained. Details of flood events were sourced from online news websites like Punch, Channels and Nigerian Agency website NiMet and interpolated into the earlier obtained weather data, ensuring reliability and consistency.

```
In [1]:
import pandas as pd
import os

# Folder containing the CSV files
folder_path = r'C:\Users\akinj\Downloads\Lagos Weather Data'

# Listing all CSV files in the folder
csv_files = [file for file in os.listdir(folder_path) if file.endswith('.csv')]

# Initialize an empty list to hold the dataframes
dataframes = []

# Iterate through each CSV file
for file in csv_files:
    file_path = os.path.join(folder_path, file)
    df = pd.read_csv(file_path)
    dataframes.append(df)

# Concatenate all dataframes into one
df = pd.concat(dataframes, ignore_index=True)
```

Data Pre-processing and Analysis

The following steps were taken to pre-process and analyze the data:

Data Cleaning

Historical weather data from 2002 to 2024 was imported into Jupyter notebook, combined and analyzed for completeness and accuracy.

```
df
```

[5]:

	name	datetime	tempmax	tempmin	temp	feelslikemax	feelslikemin	feelslike	dew	humidity	...	severerisk	sunrise	s
0	Lagos	2002-01-01	28.6	25.6	27.2	33.8	25.6	30.7	25.2	88.7	...	NaN	2002-01-01T06:57:21	20 01T18
1	Lagos	2002-01-02	33.1	26.1	29.0	39.4	26.1	33.8	25.1	80.5	...	NaN	2002-01-02T06:57:46	20 02T18
2	Lagos	2002-01-03	34.1	25.1	29.1	35.2	25.1	31.7	23.2	74.6	...	NaN	2002-01-03T06:58:11	20 03T18
3	Lagos	2002-01-04	33.1	33.1	33.1	39.4	39.4	39.4	24.1	59.3	...	NaN	2002-01-04T06:58:35	20 04T18

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

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8676 entries, 0 to 8675
Data columns (total 35 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   name              8676 non-null   object
 1   datetime          8676 non-null   object
 2   tempmax           8085 non-null   float64
 3   tempmin           8085 non-null   float64
 4   temp              7853 non-null   float64
 5   feelslikemax      8085 non-null   float64
 6   feelslikemin      8085 non-null   float64
 7   feelslike         7853 non-null   float64
 8   dew               7853 non-null   float64
```

Irrelevant columns not needed for the analysis were dropped, same as those with an excessive number of nulls. Missing values were imputed using the median in numeric columns and forward fill (ffill) for categorical columns.

```
# Filling missing values in numeric columns with median
numeric_columns = df.select_dtypes(include=['number']).columns
df[numeric_columns] = df[numeric_columns].fillna(df[numeric_columns].median())

# Filling missing values in categorical columns with forward fill
categorical_columns = df.select_dtypes(include=['object']).columns
df[categorical_columns] = df[categorical_columns].fillna(method='ffill')

# Save the cleaned dataset with filled missing values
cleaned_file_path = r'C:\Users\akinj\Downloads\cleaned_lagos_data_filled.csv'
df.to_csv(cleaned_file_path, index=False)
```

```
df.isnull().sum()
```

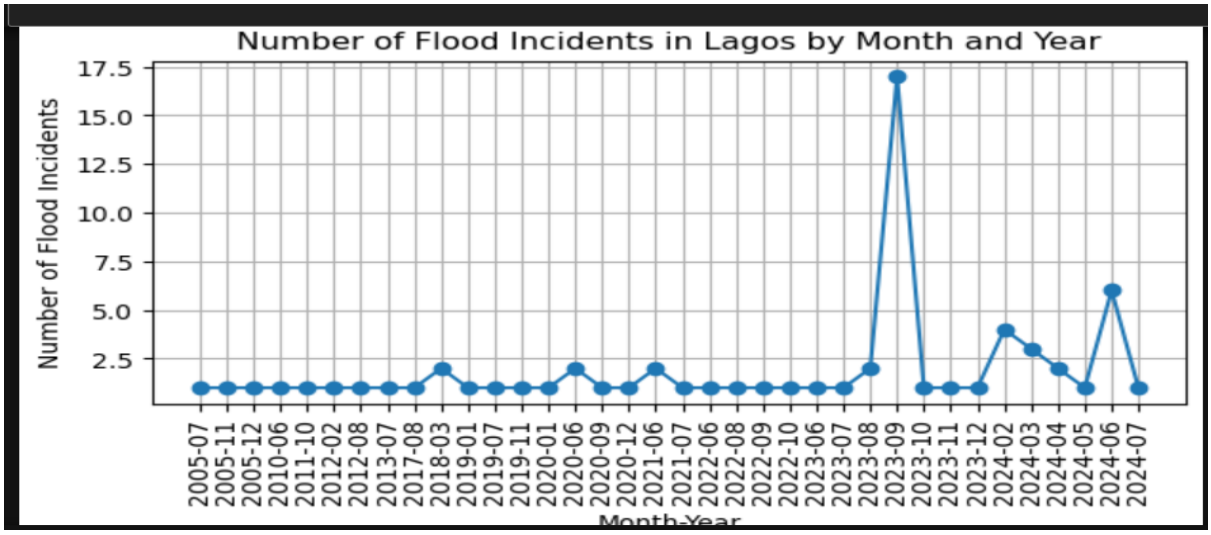
```
]: datetime          0
   tempmax           0
   tempmin           0
   temp              0
   feelslikemax      0
   feelslikemin      0
   feelslike         0
```

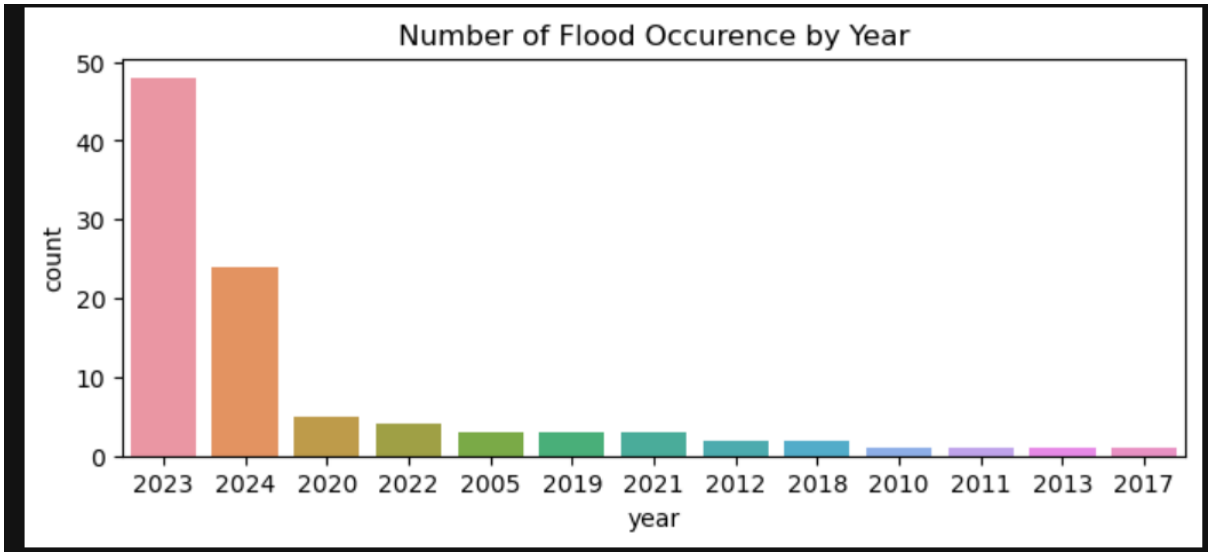
Initial Insights obtained from Dataset

Some initial insights were obtained from the cleaned Lagos flood dataset through visualizations with the aim of understanding the environmental factors that could lead to or increase the likelihood of flooding.

Flood Incidents in Lagos Over Time

Flood incidents were recorded from July 2005 to July 2024. September 2023 had the highest number of recorded flood events while June 2024 was the next highest.

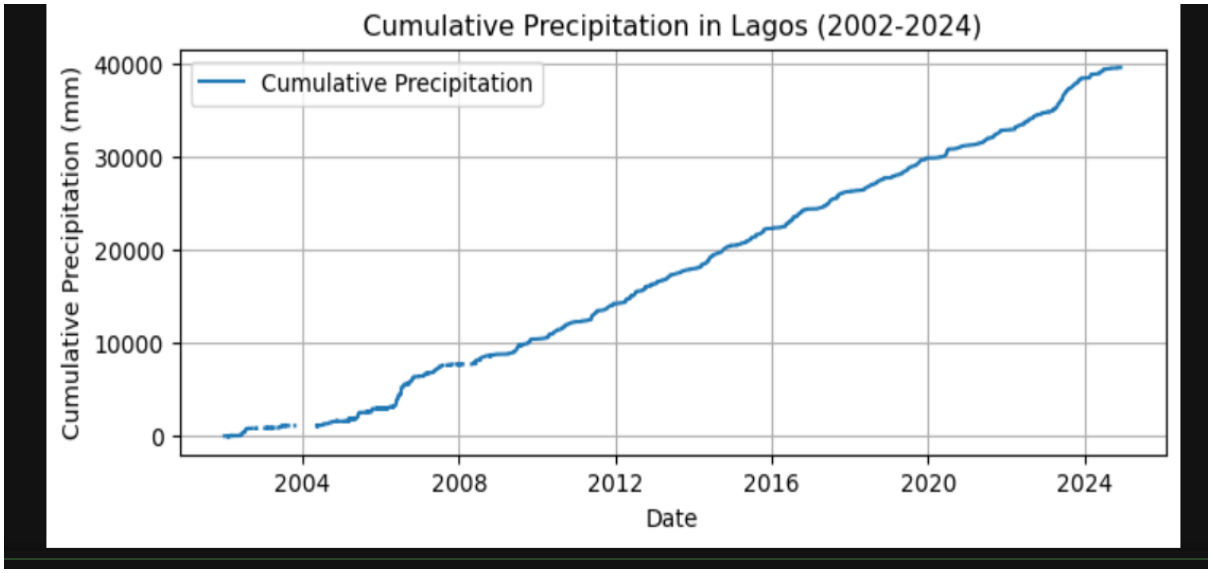




Cummulative Precipitation Over Time

The visualization shows how precipitation accumulates over time, providing a cumulative perspective on rainfall trends. It helps in assessing whether rainfall events are becoming more intense or frequent, which are critical indicators for flood risk assessment.

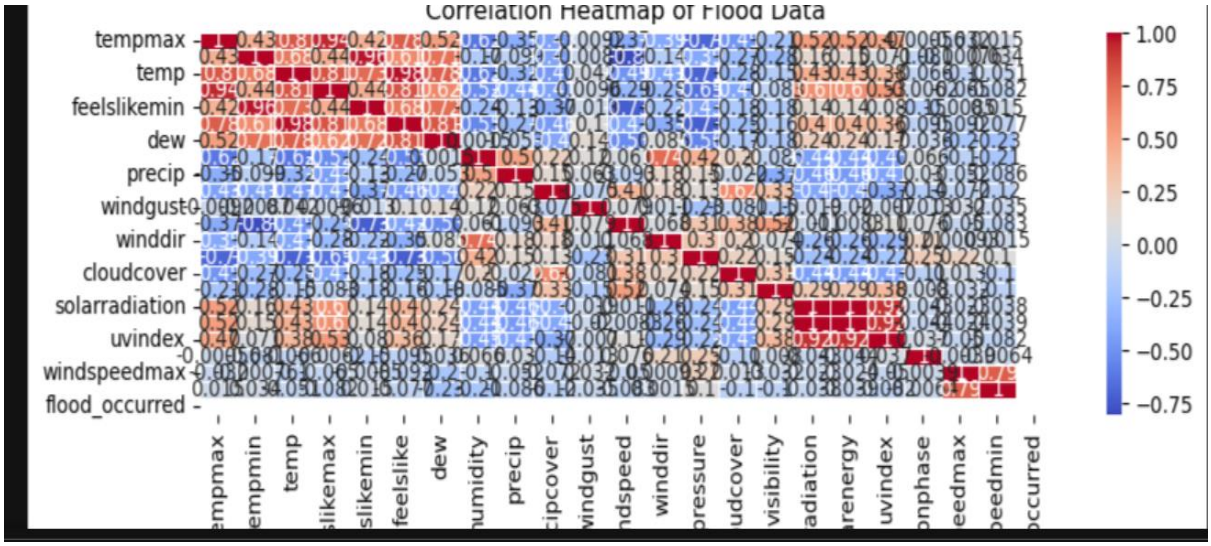
```
df['cumulative_precip'] = df['precip'].cumsum()

# Plot the cumulative precipitation over time
plt.figure(figsize=(8, 3))
plt.plot(df['datetime'], df['cumulative_precip'], label='Cumulative Precipitation')
plt.xlabel('Date')
plt.ylabel('Cumulative Precipitation (mm)')
plt.title('Cumulative Precipitation in Lagos (2002-2024)')
plt.legend()
plt.grid(True)
plt.show()

# Calculate rolling sums of precipitation over different windows
df['rolling_precip_7d'] = df['precip'].rolling(window=7).sum()
df['rolling_precip_30d'] = df['precip'].rolling(window=30).sum()
```



The chart shows us that the cumulative precipitation in Lagos has been increasing over the past few years. This suggests that the climate has changed over the past few years, and that the amount of precipitation has increased. This could be due to a number of factors, such as global warming. An increase in precipitation could lead to more flooding events.

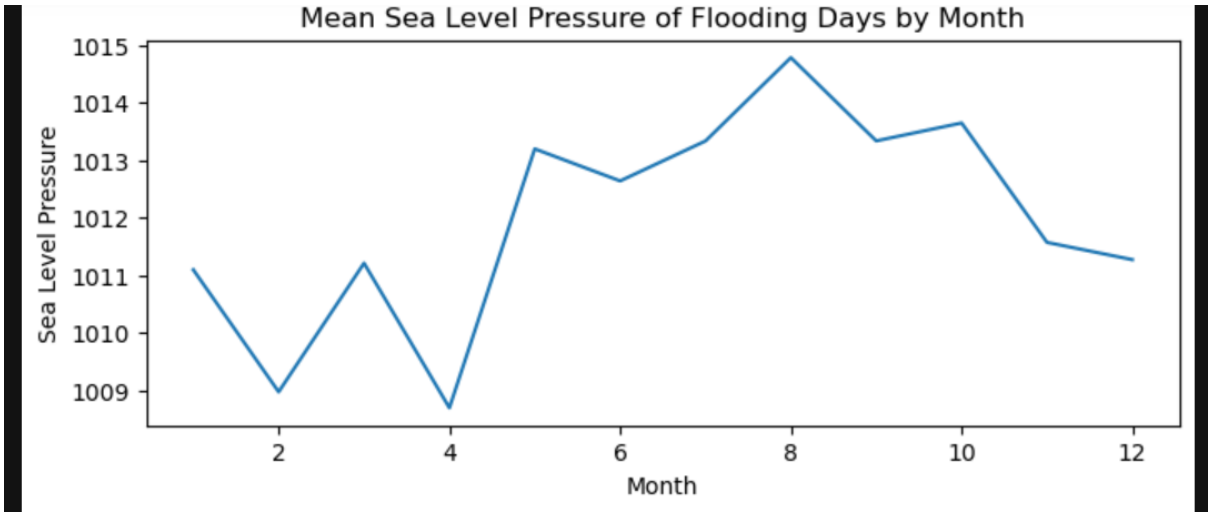
Correlation Matrix

The heatmap shows us several factors which have a strong correlation with flood occurrences. Precipitation, humidity, and cloud cover show strong positive correlations with flood occurrences, indicating they are significant predictors of flooding. In contrast, higher temperatures and wind-related variables exhibit weaker or negative correlations with flood risks.



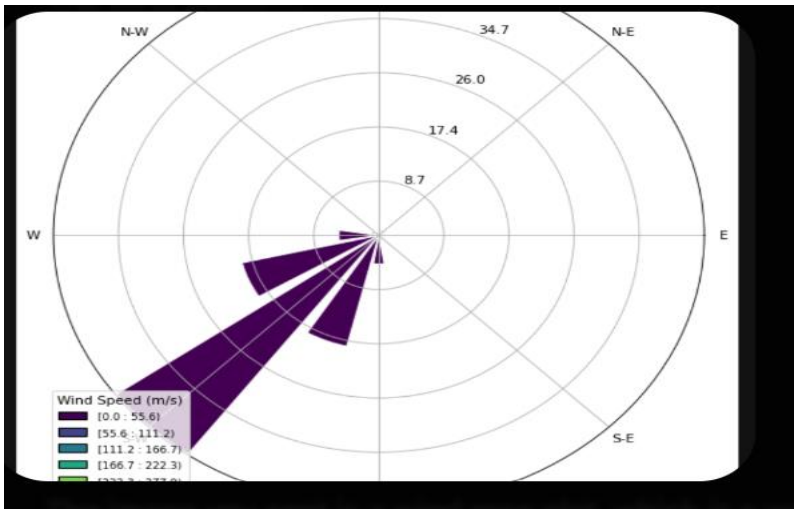
Average Sea Level Pressure by Month

The chart shows the sea level pressure peaks towards the rainy season months of June to October. Furthermore, the mean sealevel pressure, humidity and temperature were found to be 1012.5, 85.1 and 26.7 respectively.



Wind Rose Plot

The wind rose plot illustrates the distribution of wind speed and direction over a specified period. Understanding wind patterns is essential because winds can influence the movement of weather systems, the intensity of precipitation and the distribution of moisture. For example, winds from certain directions might bring in moisture-laden air that increases precipitation and flood risk in specific areas.

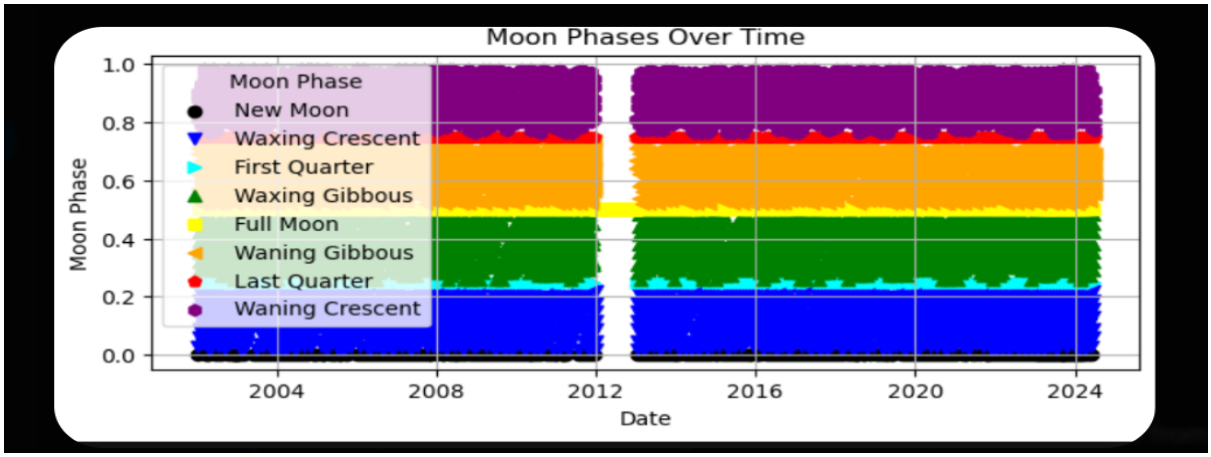


The wind rose plot shows that the prevailing wind direction in Lagos is northerly. This means that the wind blows from the north more often than from any other direction.

Wind speed: The wind speed is strongest from the north and weakest from the south. This information could be helpful for developing flood prediction models that take into account wind direction and speed.

Moon Cycle Chart

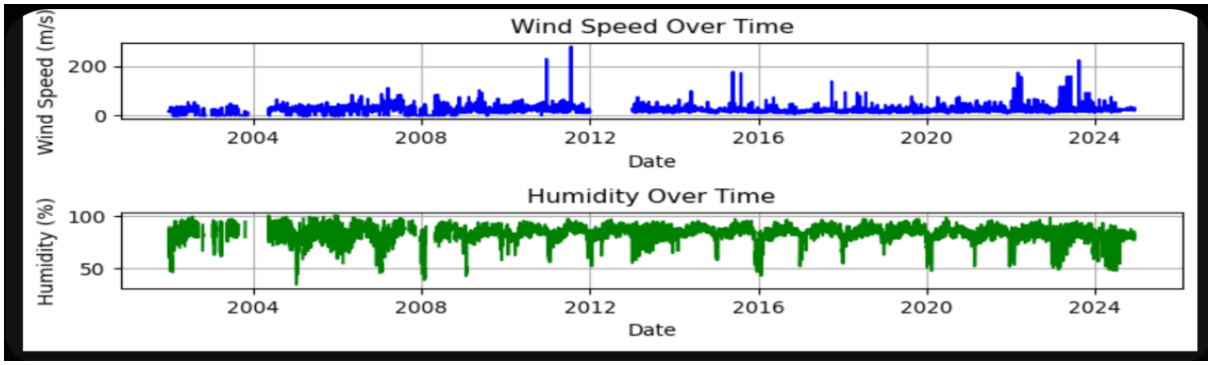
Tracking the moon cycle can be insightful because lunar phases can influence tidal patterns and potentially affect water levels in coastal areas or river systems. This visualization helps in understanding how lunar cycles correlate with flood events, especially in regions where tidal influences play a significant role.



The chart shows the eight phases of the moon and their relation to the Earth over a 20 year period including the new moon, waxing crescent, first quarter, waxing gibbous, full moon, waning gibbous, last quarter, and waning crescent. It also appears the full moon phase repeats roughly every 29 days.

Wind Speed and Humidity Chart

The wind speed and humidity chart helps in understanding the atmospheric conditions contributing to flooding. The wind speed chart helps identify storm intensity, patterns, and potential severe weather indicators while the humidity chart reveals moisture availability and its correlation with heavy rainfall. The charts provide a comprehensive view of the factors which lead to floods in Lagos, enhancing predictive modeling accuracy and supporting early warning systems while indicating critical changes in weather conditions.



Merging Flood Dataset with Cleaned Weather Dataset

After obtaining initial insights, the weather dataset was prepared for predictive modelling by merging with the dataset containing flood incidents on the datetime column. A new column, 'flood_occurred' was generated to denote whether a flood transpired on each date, converting the presence of a flood event into binary format (1 for occurrence and 0 for none). This new column served as the target, predictive variable for our model.

```
# Loading the weather dataset
weather_df = pd.read_csv(r'C:\Users\akinj\Downloads\cleaned_lagos_data_filled.csv')
weather_df['datetime'] = pd.to_datetime(weather_df['datetime'])

# Loading the flood dates dataset from an Excel file
flood_dates_df = pd.read_excel(r'C:\Users\akinj\Downloads\lagos_flood_data.xlsx')
flood_dates_df['Date'] = pd.to_datetime(flood_dates_df['Date'])

# Merging the two datasets
merged_df = weather_df.merge(flood_dates_df, left_on='datetime', right_on='Date', how='left')

# Creating a new column to indicate whether a flood occurred on that date
merged_df['flood_occurred'] = merged_df['Date'].notna().astype(int)

# Dropping the Date column after extraction of month, day, year
```

Feature Selection:

Key weather attributes such as temperature (max, min, average), humidity, precipitation, wind speed, sea level pressure, cloud cover, visibility, and date-related features (year, month, day, day of the week) were selected as features for analysis.

```
df['month'] = df['datetime'].dt.month
df['day'] = df['datetime'].dt.day
df['day_of_week'] = df['datetime'].dt.dayofweek

# Dropping the original datetime column
df.drop(columns=['datetime'], inplace=True)

# Defining feature columns
feature_columns = ['tempmax', 'tempmin', 'temp', 'humidity', 'precip', 'windspeed', 'sealevelpressure',
                   'cloudcover', 'visibility', 'year', 'month', 'day', 'day_of_week']

# Filling missing values with the median of each column
```

Flood Prediction Model

Model Selection

A machine learning approach was chosen for flood prediction due to its ability to handle complex patterns and relationships in the data. Various models were evaluated, including Logistic Regression, Random Forest, and XGBoost, with Random Forest and XGBoost showing the highest accuracy.

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
import joblib

# Splitting data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initializing models
logistic_model = LogisticRegression()
rf_model = RandomForestClassifier(random_state=42)
xgb_model = XGBClassifier(random_state=42)

# Training models
logistic_model.fit(X_train, y_train)
rf_model.fit(X_train, y_train)
xgb_model.fit(X_train, y_train)

# Evaluating models
logistic_train_accuracy = logistic_model.score(X_train, y_train)
logistic_test_accuracy = logistic_model.score(X_test, y_test)
print(f"Logistic Regression Training accuracy: {logistic_train_accuracy:.2f}")
```

Model Development Process

Training and Validation: The models were trained on the historical weather data, with a focus on balancing the dataset due to the low number of flood instances.

SMOTE Resampling: This was applied to address the class imbalance, enhancing the model's ability to predict flood occurrences accurately.

```
Logistic Regression Training accuracy: 0.99
Logistic Regression Testing accuracy: 0.99
Random Forest Training accuracy: 1.00
Random Forest Testing accuracy: 1.00
XGBoost Training accuracy: 1.00
XGBoost Testing accuracy: 1.00
Logistic Regression model saved to logistic_regression_model.pkl
Random Forest model saved to random_forest_model.pkl
XGBoost model saved to xgboost_model.pkl
```

```
[8]: from imblearn.over_sampling import SMOTE

     # Defining the feature columns and target variable
     X = df[feature_columns]
     y = df[target]

     # Applying SMOTE to balance the dataset
     smote = SMOTE(random_state=42)
     X_resampled, y_resampled = smote.fit_resample(X, y)

     # Splitting the resampled data into training and testing sets
     X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2, random_state=42)
```

Hyperparameter Tuning

Hyperparameters for the XGBoost model were fine-tuned using Grid Search, leading to improved performance and reduced overfitting.

```
# Initializing XGBClassifier
xgb_model = XGBClassifier(random_state=42)

# Performing Grid Search
grid_search = GridSearchCV(estimator=xgb_model, param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

# Getting best parameters and best score
print("Best Parameters:", grid_search.best_params_)
print("Best Training Accuracy:", grid_search.best_score_)

# Using best model
best_xgb_model = grid_search.best_estimator_
```

```
Best Parameters: {'colsample_bytree': 0.6, 'learning_rate': 0.5, 'max_depth': 7, 'min_child_weight': 1, 'subsample': 1.0}
Best Training Accuracy: 0.9952586206896552
```

```
best_xgb_model.fit(X_train, y_train)

# Evaluating the model
train_accuracy = best_xgb_model.score(X_train, y_train)
test_accuracy = best_xgb_model.score(X_test, y_test)

print(f"Training Accuracy: {train_accuracy:.4f}")
print(f"Testing Accuracy: {test_accuracy:.4f}")
```

```
Training Accuracy: 1.0000
Testing Accuracy: 0.9977
```

Testing the Flood Prediction Model with ipywidgets

The model was tested with ipywidgets, an interactive HTML widget for Jupyter notebooks. Using a defined function that takes weather data inputs and processes them, the trained model was used to make a flood prediction. With future weather data for Lagos State from July 6th to 11th, the model predicted flooding would occur on the 11th of July 2024.

```
model = joblib.load(model_path)

# Providing future weather data
data = {
    'datetime': ['07/05/2024', '07/06/2024', '07/07/2024', '07/08/2024', '07/09/2024', '07/10/2024', '07/11/2024'],
    'tempmax': [30, 27.3, 26.9, 26.3, 26.9, 25.9, 25.6],
    'tempmin': [24, 26.5, 26, 25.2, 24.8, 25.1, 23.8],
    'temp': [27.4, 26.8, 26.4, 25.7, 25.9, 25.5, 24.6],
    'humidity': [83.1, 80.1, 80.6, 84.4, 83, 84.9, 90.1],
    'precip': [3.3, 5.9, 5.9, 24, 16, 27.1, 100],
    'windspeed': [22.3, 27.7, 29.9, 23.8, 31.7, 28.4, 25.2],
    'sealevelpressure': [1014.1, 1012.7, 1012.9, 1012.8, 1012.3, 1013.3, 1014.1],
    'cloudcover': [49.2, 97.1, 99.6, 99.9, 82.9, 95.8, 99.9],
    'visibility': [11.3, 24.1, 20.9, 13.6, 17.5, 9.7, 6.4],
```

```
# Adding predictions to the DataFrame
future_weather_data['flood_prediction'] = predictions
future_weather_data['flood_prediction'] = future_weather_data['flood_prediction'].apply(lambda x: 'Flood predicted' i

# Displaying the predictions
print(future_weather_data[['datetime', 'flood_prediction']])
```

```
    datetime    flood_prediction
0 2024-07-05  No flood predicted
1 2024-07-06  No flood predicted
2 2024-07-07  No flood predicted
3 2024-07-08  No flood predicted
4 2024-07-09  No flood predicted
5 2024-07-10  No flood predicted
6 2024-07-11     Flood predicted
```

The prediction can be considered valid, especially since the forecasted temperature, humidity and sea level pressure fall within the range of past flood incidents in our data, as indicated by the mean calculations. Key factors influencing flood predictions include precipitation, humidity, and wind speed. These variables showed strong correlations with flood occurrences, as identified through our earlier exploratory data analysis.

Limitations

Some limitations to this study and model building include data imbalance due to the limited number of recorded flood instances; there were 8000+ rows of weather data compared to less than 100 records of flood data. This imbalance also led to overfitting which I tried to mitigate through resampling with SMOTE and hyperparameter finetuning.

Conclusion

This report investigated the development of a flood prediction model for Lagos, Nigeria, using historical data on precipitation, wind speed, humidity, and moon phases. Despite limitations such as data imbalance, the analysis successfully developed a flood prediction model for Lagos, achieving acceptable accuracy and reliability.

Key predictors identified include precipitation, humidity, and cloud cover, which showed strong correlations with flood occurrences. The model's predictions can significantly enhance flood risk management and disaster preparedness in Lagos.

Future Work

Areas that can be looked into to improve flood mitigation in Lagos state include:

Increasing Data Volume: Collecting more flood data instances to improve model training.

Model Enhancement: Exploring additional features and advanced modeling techniques to further improve prediction accuracy.

Thank you.

References

https://www.bbc.com/pidgin/articles/czvxdzwr97go

https://www.thecable.ng/photos-commuters-stranded-homes-submerged-as-flood-ravages-lagos/

https://www.visualcrossing.com/weather/weather-data-services

https://nimet.gov.ng/

Code Notebook Link:

https://drive.google.com/file/d/1TPiTB5h4WCLmpp5ewd9eVwShr4uBG5Nk/view?usp=sharing