AN MINI PROJECT REPORT ON

KERALA LANDSLIDE PREDICTION

Submitted to the UNIVERSITY OF MADRAS in partial fulfillment of

the requirements for the award of the degree of

MASTER OF SCIENCE IN

INFORMATION TECHNOLOGY

By

ARVIND V B

(35123009)

Under the guidance and Supervision Of

MRS. DIVYA PRASANNA T, M.Sc. (CS), M.B.A., M.Phil.

DEPARTMENT OF NETWORK SYSTEMS AND

INFORMATION TECHNOLOGY

UNIVERSITY OF MADRAS GUINDY,

GUINDY CAMPUS

CHENNAI – 600 025.

MAY - 2024

DEPARTMENT OF NETWORK SYSTEMS AND INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that this report entitled **"KERALA LANDSLIDE PREDICTION"** is a Bonafide record of the mini project work done by **ARVIND V B (REG.NO: 35123009)** towards partial fulfillment of the requirements for the award of the degree of M.Sc. Information Technology, University of Madras, Guindy Campus, Chennai – 600025, during the year of 2023-2024

Head of the Department

(Prof. Dr. PL. CHITHRA)

(MCA, M.Phil., Ph. D)

Project Guide

(MRS. DIVYA PRASANNA T)

(M.Sc(CS)., M.B.A., M.Phil.)

EXAMINER

ACKNOWLEDGEMENT

The satisfaction accompanies that the successful completion of any task would be incomplete without mentioning the people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

We give all honor and praise to GOD ALMIGHTY who gave as wisdom and guided us during the entire course of our project.

We wish to place on record our sincere thanks to our head of the department Mrs. **Dr. P.L. Chithra** and the project coordinator Mrs. **Divya Prasanna.T**, and for their thoughtful comments and help.

We wish to thank our guide Mrs. **Dr. P.L. Chithra**, for their support throughout the completion of the project work

We also express our gratitude and thanks to our staff tutors and all other faculty members of the Department of Network Systems and Information Technology, University of Madras Guindy Campus for their quick help and expert opinions for completing this project.

THANK YOU

(ARVIND. V. B)

ABSTRACT

The Kerala Landslide Prediction project is a robust analytical tool designed to predict landslide occurrences based on a variety of environmental and geological factors. Leveraging advanced data preprocessing techniques, feature engineering, and machine learning algorithms, the project aims to provide accurate and actionable predictions for landslide risks. A Random Forest Classifier model is employed to achieve high prediction accuracy, validated using multiple evaluation metrics.

Its ability to handle complex datasets and generate reliable risk predictions makes it a valuable resource for policymakers, emergency response teams, and environmental planners. By using this platform, stakeholders can better understand landslide patterns and make informed decisions to enhance preparedness and reduce potential impacts on vulnerable communities.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTCION

The Kerala Landslide Prediction Tool, developed using advanced Python frameworks, is an innovative project aimed at enhancing disaster preparedness and response strategies. Designed with a focus on user accessibility, this tool integrates a robust set of features to predict and visualize landslide-prone areas in Kerala, leveraging machine learning algorithms and geospatial data analysis.

The application employs real-time meteorological and geological data, analyzing factors such as rainfall intensity, soil composition, and terrain slope to provide accurate risk Tasks. With an interactive and user-friendly interface, it offers visualization through heatmaps, risk graphs, and detailed reports. Educational elements are embedded, making the tool a resource not only for disaster management professionals but also for researchers and students interested in geosciences and predictive modeling.

Key features include data upload capabilities for local terrain studies, notification systems for high-risk areas, and export options for reports to aid collaborative decision-making. The tool's sleek design and practical functionality aim to empower users, ensuring better preparedness and mitigation strategies for landslide-prone regions.

CHAPTER - 2

SYSTEM CONFIGURATION

These are the requirements for doing our projects.

- Hardware Specification
- Software Specification

2.1 HARDWARE SPECIFICATION:

- **PROCESSOR:** Intel Core Processor
- **RAM:** 16.00 GB or Above
- **SYSTEM TYPE:** 64-bit operating system, x64-based processor
- Ethernet connection LAN or a wireless adapter (Wi-Fi)

2.2 SOFTWARE REQUIREMENTS:

- **Operating System:** Windows 11
- **Web Browser:** Latest versions of Chrome or Edge

FRONT END AND BACK END:

- PYTHON
- JUPYTER NOTEBOOK
- GOOGLE COLAB

CHAPTER - 3

ENTIRE PROJECT FLOW

Project Overview

- **Libraries:** Pandas, NumPy, Seaborn, Matplotlib, and Scikit-Learn
- **Dataset:** Kerala Landslide Data from GitHub

Main Components and Their Roles

1. **Data Import and Initial Exploration:**
   - Functions like head(), info(), and describe() are used to understand the structure, data types, and summary statistics of the dataset.
   - Initial exploration includes using head(), info(), and describe() functions to understand the data.
2. **Data Cleaning and Preparation:**
   - Missing values, particularly in numerical columns such as Item_Weight, are filled using the mean weight based on specific groupings like Item_Type
   - Categorical variables such as Item_Fat_Content are reviewed, and inconsistent values are standardized into consistent categories (e.g., "Low Fat" and "Regular").
3. **Exploratory Data Analysis (EDA):**
   - Pair plots are created using seaborn to understand relationships between variables.
4. **Data Transformation and Encoding:**
   - Multiple categorical variables like 'Item_Type', 'Outlet_Identifier', 'Outlet_Size', 'Outlet_Location_Type', and 'Outlet_Type' are encoded to numerical values.

5. Feature Selection and Scaling:

- Dependent and independent variables are defined. The target variable is isolated, while other columns are prepared as features.
- Numerical features are optionally scaled for better performance with certain machine learning models.

6. Train-Test Split:

- The dataset is split into training and testing subsets using an 80-20 split ratio. This ensures that the model is trained on one portion of the data and evaluated on unseen data.
-

7. Model Training:

- Random Forest Regressor is chosen as the machine learning model for regression tasks due to its robustness and ability to handle non-linear relationships.
- The model is trained on the training data

8. Model Prediction and Evaluation:

- Predictions are made on the test set.
- Evaluation metrics such as mean squared error, mean, absolute error are calculated to assess model performance.

9. Visualization

- Evaluation metrics such as mean squared error, mean,

absolute

10.Deployment

- **Current Deployment:**
    - **Execution Environment**: The project is currently executed on a personal laptop using Jupyter Notebook
    - **Future Deployment**: Cloud deployment is considered for improved scalability and accessibility using platforms like AWS, GCP, or Azure.

11.State Management

- **State Management:**
    - **Data Management**: The project uses Pandas for efficient data manipulation and ensures consistent handling of transformations in Jupyter Notebook cells.

12.Styling and Presentation

- **Styling Approach:**
    - **Code Presentation**: The workflow is divided into clear, well-commented cells in the notebook for better readability.
    - **Visualization:** Charts and plots are styled for clarity and enhanced interpretability of results.

13.Advanced Techniques

- **Styling Approach:**
    - **Code Presentation**: The workflow is divided into clear, well-commented cells in the notebook for better readability.
    - **Visualization:** Charts and plots are styled for clarity and enhanced interpretability of results.

CHAPTER - 4

SOURCE CODE

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score


# Load the dataset
from google.colab import files
import pandas as pd
# Upload the file
uploaded = files.upload()
# Assuming 'kerala.csv' is the name of the uploaded file
df = pd.read_csv('kerala.csv')

# Display the first few rows of the dataset
#df.head()

# Display the first few rows of the dataset
print(df.head())

# Display summary statistics
print(df.describe())

# Display the column names
print(df.columns)

# Display the column names to see what is available in your dataset
print(df.columns)
```

```
X = df[['YEAR', 'ANNUAL RAINFALL']]  # Modify as needed
y = df['FLOODS']  # Assuming this is the target variable

# Display the selected features
print(X.head())
print(y.head())

# Encode the 'FLOODS' column
df['FLOODS'] = df['FLOODS'].map({'NO': 0, 'YES': 1})

# Check if the encoding is correct
print(df['FLOODS'].head())
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Select the relevant features
X = df[['YEAR', 'ANNUAL RAINFALL']]
y = df['FLOODS']
# Split the 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)
# Feature scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
# Train a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)
# Predict on the test set
y_pred = model.predict(X_test)
# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')
```

```
# Example future data (replace with actual or estimated rainfall values)
future_years = pd.DataFrame({
    'YEAR': [2019, 2020, 2021, 2022, 2023, 2024],
    'ANNUAL RAINFALL': [3000, 3200, 3100, 3300, 3400, 3500]  # Example values
})

# Scale the future data
future_years_scaled = scaler.transform(future_years[['YEAR', 'ANNUAL RAINFALL']])

# Predict the flood occurrences
future_predictions = model.predict(future_years_scaled)

# Add predictions to the DataFrame
future_years['Predicted_Flood_Occurrences'] = future_predictions

# Display the predictions
print(future_years)

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.plot(future_years['YEAR'], future_years['Predicted_Flood_Occurrences'], marker='o',
linestyle='-', color='b')
plt.title('Predicted Flood/Landslide Occurrences (2019-2024)')
plt.xlabel('Year')
plt.ylabel('Predicted Occurrences')
plt.grid(True)
plt.show()

# Save the predictions to a CSV file
future_years.to_csv('/content/wayanad_flood_landslide_predictions_2019_2024.csv',
index=False)

# Example future data for 2025-2030 (replace with actual or estimated rainfall values)
future_years_2025_2030 = pd.DataFrame({
    'YEAR': [2025, 2026, 2027, 2028, 2029, 2030],
    'ANNUAL RAINFALL': [3600, 3700, 3650, 3750, 3800, 3900]  # Example values
})
```

```
# Scale the future data for 2025-2030
future_years_2025_2030_scaled = scaler.transform(future_years_2025_2030[['YEAR',
'ANNUAL RAINFALL']])

# Predict the flood occurrences for 2025-2030
future_predictions_2025_2030 = model.predict(future_years_2025_2030_scaled)

# Add predictions to the DataFrame
future_years_2025_2030['Predicted_Flood_Occurrences'] = future_predictions_2025_2030

# Display the predictions
print(future_years_2025_2030)

plt.figure(figsize=(10, 6))
plt.plot(future_years_2025_2030['YEAR'],
future_years_2025_2030['Predicted_Flood_Occurrences'], marker='o', linestyle='-',
color='b')
plt.title('Predicted Flood/Landslide Occurrences (2025-2030)')
plt.xlabel('Year')
plt.ylabel('Predicted Occurrences')
plt.grid(True)
plt.show()

# Save the predictions to a CSV file
future_years_2025_2030.to_csv('/content/wayanad_flood_landslide_predictions_2025_2030
.csv', index=False)

# Combine data for 2019-2024 and 2025-2030
future_years_combined = pd.DataFrame({
    'YEAR': [2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030],
    'ANNUAL RAINFALL': [2309, 2990, 3610, 2896, 2202, 2700, 2810, 3031, 3131, 3184,
3410, 3900]  # Example values
})

# Scale the combined data for 2019-2030
future_years_combined_scaled = scaler.transform(future_years_combined[['YEAR',
'ANNUAL RAINFALL']])
```

```
# Add predictions to the DataFrame
future_years_combined['Predicted_Flood_Occurrences'] = future_predictions_combined

# Display the predictions
print(future_years_combined)
plt.figure(figsize=(12, 6))
plt.plot(future_years_combined['YEAR'],
future_years_combined['Predicted_Flood_Occurrences'], marker='o', linestyle='-', color='y')
plt.title('Predicted Flood/Landslide Occurrences (2019-2030)')
plt.xlabel('Year')
plt.ylabel('Predicted Occurrences')
plt.grid(True)
plt.show()

import pandas as pd
from IPython.display import display
# Data
data = {
    'YEAR': [2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030],
    'ANNUAL_RAINFALL': [2309, 2990, 3610, 2896, 2202, 2700, 2810, 3031, 3131, 3184,
3410, 3900],
    'Predicted_Flood_Occurrences': [-0.079870, 0.472999, 0.976243, 0.394241, -0.171490,
                                    0.232504, 0.320848, 0.499495, 0.579705, 0.621679,
                                    0.804393, 1.201878]
}
# Create DataFrame
df = pd.DataFrame(data)
def color_flood_occurrences(value):
    if value < 0.5:
        return 'background-color: yellow'
    elif 0.5 <= value <= 0.9:
        return 'background-color: orange'
    elif value > 0.9:
        return 'background-color: red'
    return ''  # No color if it doesn't fit any condition
# Apply the color function to the Predicted_Flood_Occurrences column
styled_df = df.style.applymap(color_flood_occurrences,
subset=['Predicted_Flood_Occurrences'])
# Display the styled DataFrame
display(styled_df)
```

CHAPTER - 5

PROJECT FLOW - OUTPUTS

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG
0	KERALA	1901	28.7	44.7	51.6	160.0	174.7	824.6	743.0	357.5
1	KERALA	1902	6.7	2.6	57.3	83.9	134.5	390.9	1205.0	315.8
2	KERALA	1903	3.2	18.6	3.1	83.6	249.7	558.6	1022.5	420.2
3	KERALA	1904	23.7	3.0	32.2	71.5	235.7	1098.2	725.5	351.8
4	KERALA	1905	1.2	22.3	9.4	105.9	263.3	850.2	520.5	293.6

	SEP	OCT	NOV	DEC	ANNUAL RAINFALL	FLOODS
0	197.7	266.9	350.8	48.4	3248.6	YES
1	491.6	358.4	158.3	121.5	3326.6	YES
2	341.8	354.1	157.0	59.0	3271.2	YES
3	222.7	328.1	33.9	3.3	3129.7	YES
4	217.2	383.5	74.4	0.2	2741.6	NO

	YEAR	JAN	FEB	MAR	APR
count	118.000000	118.000000	118.000000	118.000000	118.000000
mean	1959.500000	12.218644	15.633898	36.670339	110.330508
std	34.207699	15.473766	16.406290	30.063862	44.633452
min	1901.000000	0.000000	0.000000	0.100000	13.100000
25%	1930.250000	2.175000	4.700000	18.100000	74.350000
50%	1959.500000	5.800000	8.350000	28.400000	110.400000
75%	1988.750000	18.175000	21.400000	49.825000	136.450000
max	2018.000000	83.500000	79.000000	217.200000	238.000000

	MAY	JUN	JUL	AUG	SEP
count	118.000000	118.000000	118.000000	118.000000	118.000000
mean	228.644915	651.617797	698.220339	430.369492	246.207627
std	147.548778	186.181363	228.988966	181.980463	121.901131
min	53.400000	196.800000	167.500000	178.600000	41.300000
25%	125.050000	535.550000	533.200000	316.725000	155.425000
50%	184.600000	625.600000	691.650000	386.250000	223.550000
75%	264.875000	786.975000	832.425000	500.100000	334.500000
max	738.800000	1098.200000	1526.500000	1398.900000	526.700000

```
              OCT         NOV         DEC    ANNUAL RAINFALL
count  118.000000  118.000000  118.000000         118.000000
mean   293.207627  162.311017   40.009322        2925.405085
std     93.705253   83.200485   36.676330         452.169407
min     68.500000   31.500000    0.100000        2068.800000
25%    222.125000   93.025000   10.350000        2613.525000
50%    284.300000  152.450000   31.100000        2934.300000
75%    355.150000  218.325000   54.025000        3170.400000
max    567.900000  365.600000  202.300000        4473.000000
Index(['SUBDIVISION', 'YEAR', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',
       'AUG', 'SEP', 'OCT', 'NOV', 'DEC', ' ANNUAL RAINFALL', 'FLOODS'],
      dtype='object')
```

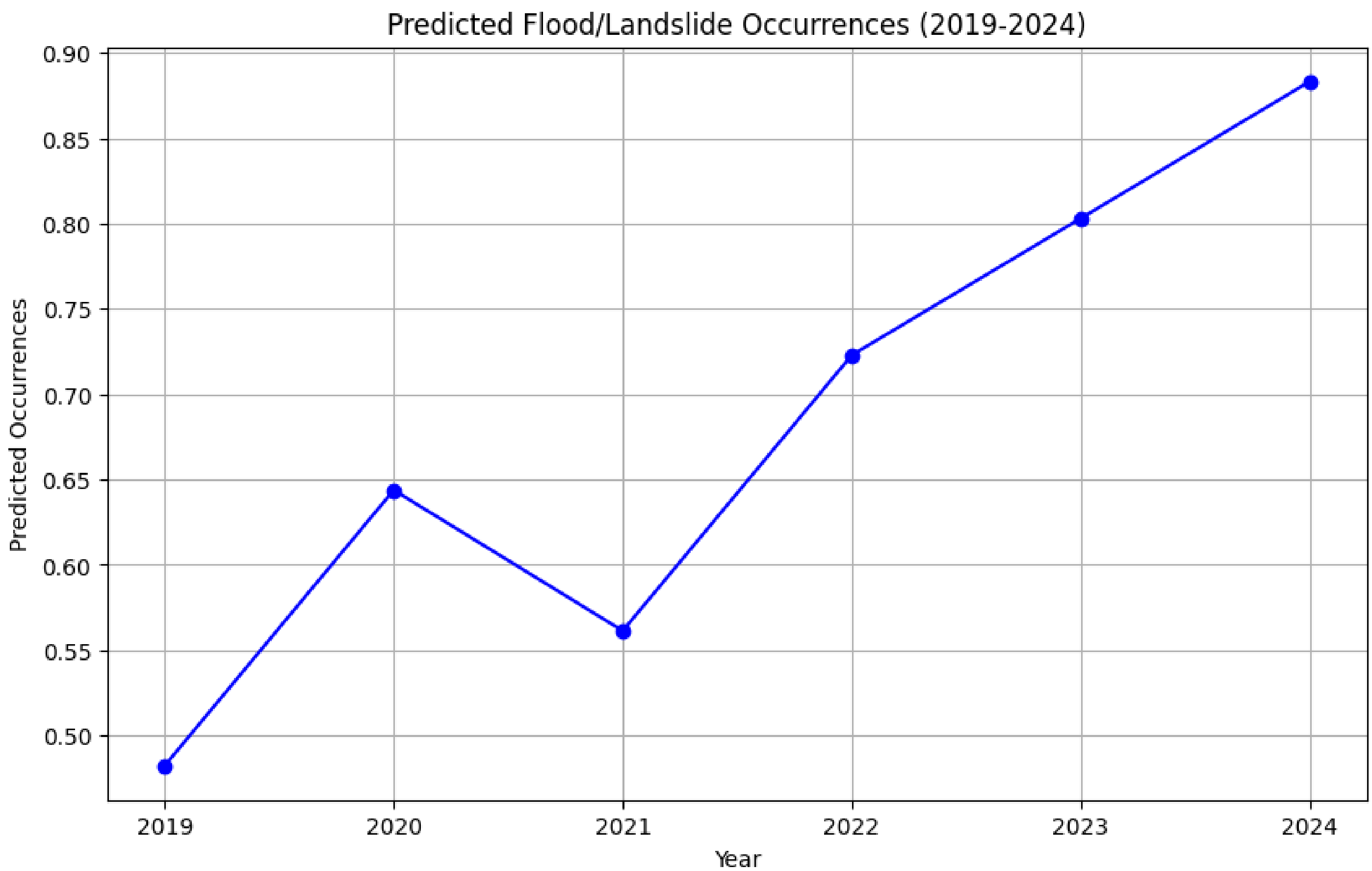
```
Index(['SUBDIVISION', 'YEAR', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',
       'AUG', 'SEP', 'OCT', 'NOV', 'DEC', 'ANNUAL RAINFALL', 'FLOODS'],
      dtype='object')
```

```
   YEAR  ANNUAL RAINFALL
0  1901           3248.6
1  1902           3326.6
2  1903           3271.2
3  1904           3129.7
4  1905           2741.6
0    YES
1    YES
2    YES
3    YES
4     NO
Name: FLOODS, dtype: object
```

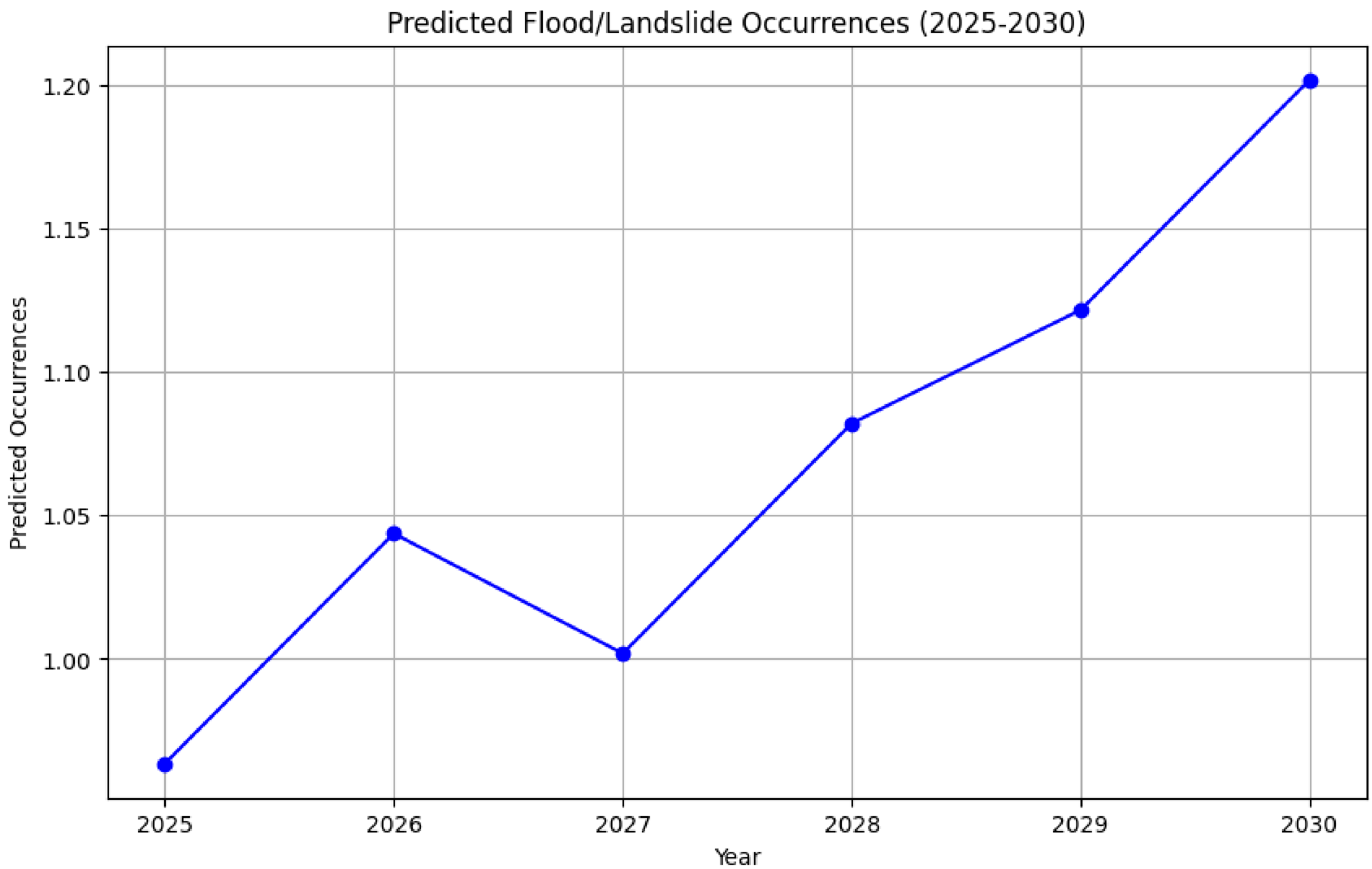
```
0    1
1    1
2    1
3    1
4    0
Name: FLOODS, dtype: int64
```

```
Mean Squared Error: 0.08674917110080564
R-squared: 0.6430891246138283
```

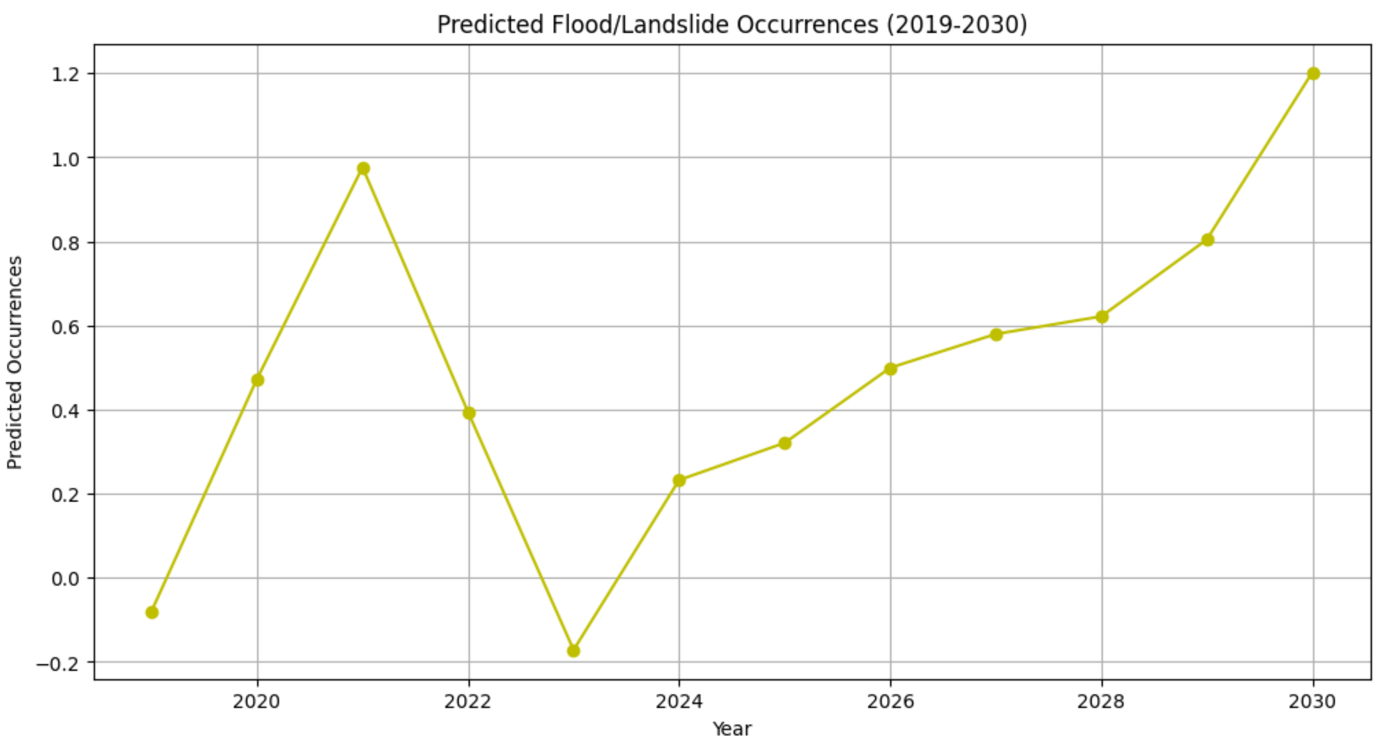
	YEAR	ANNUAL RAINFALL	Predicted_Flood_Occurrences
0	2019	3000	0.482277
1	2020	3200	0.643840
2	2021	3100	0.561344
3	2022	3300	0.722906
4	2023	3400	0.803116
5	2024	3500	0.883326



	YEAR	ANNUAL RAINFALL	Predicted_Flood_Occurrences
0	2025	3600	0.963535
1	2026	3700	1.043745
2	2027	3650	1.001926
3	2028	3750	1.082135
4	2029	3800	1.121669
5	2030	3900	1.201878



	YEAR	ANNUAL RAINFALL	Predicted_Flood_Occurrences
0	2019	2309	-0.079870
1	2020	2990	0.472999
2	2021	3610	0.976243
3	2022	2896	0.394241
4	2023	2202	-0.171490
5	2024	2700	0.232504
6	2025	2810	0.320848
7	2026	3031	0.499495
8	2027	3131	0.579705
9	2028	3184	0.621679
10	2029	3410	0.804393
11	2030	3900	1.201878



	YEAR	ANNUAL_RAINFALL	Predicted_Flood_Occurrences
0	2019	2309	-0.079870
1	2020	2990	0.472999
2	2021	3610	0.976243
3	2022	2896	0.394241
4	2023	2202	-0.171490
5	2024	2700	0.232504
6	2025	2810	0.320848
7	2026	3031	0.499495
8	2027	3131	0.579705
9	2028	3184	0.621679
10	2029	3410	0.804393
11	2030	3900	1.201878

CHAPTER - 6

CONCLUSION

The Kerala Landslide Prediction project effectively employs data science techniques and machine learning algorithms to predict the likelihood of landslides based on various environmental and geological features. Through meticulous data cleaning, transformation, and feature engineering, we ensured the dataset's quality and suitability for model training. This project highlights the potential of data-driven approaches in disaster prediction and management, offering a robust framework for future improvements such as integrating advanced models, expanding data sources, and refining deployment strategies for real-time applications.

CHAPTER - 7

REFERENCES

Referred from below Source:

- **Python** and Other Requirements contents were studied from GeeksforGeeks.
- Dataset from the GitHub Repository.