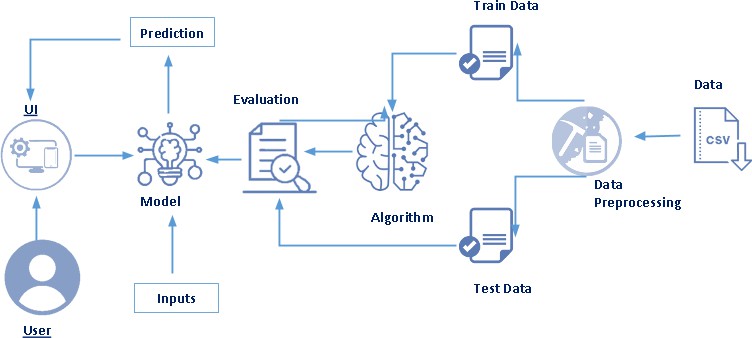


Flood Prediction Using Machine Learning

Floods are among the most devastating natural disasters globally, causing significant loss of life, property damage, and disruption to infrastructure and economies. Accurate and timely flood prediction is crucial for issuing warnings, enabling effective disaster management, and minimizing adverse impacts. Traditional flood prediction methods often rely on hydrological models which can be complex and require extensive data. Machine Learning offers a powerful alternative or complementary approach by identifying patterns and relationships in various influencing factors like rainfall, river levels, topography, and historical data to predict flood events.

However, predicting floods accurately is challenging due to the complex interplay of factors and the inherent randomness of weather patterns. The main purpose of this Flood Prediction system is to predict the likelihood or timing of a flood event based on different environmental parameters and historical patterns.

**Technical Architecture:**

****

Project Flow:

* Data related to flood influencing factors (e.g., rainfall, river levels, soil moisture, historical events) is collected.
* This data is pre-processed and used to train a machine learning model.
* The trained model analyzes current conditions.
* Based on the analysis, the model predicts the likelihood or severity of a flood event, which can then be used to issue warnings or inform response strategies.

To accomplish this, we have to complete all the activities listed below,

* Define Problem / Problem Understanding
  + Specify the business problem
  + Business requirements
  + Literature Survey
  + Social or Business Impact.
* Data Collection & Preparation
  + Collect the dataset
  + Data Preparation
* Exploratory Data Analysis
  + Descriptive statistical
  + Visual Analysis
* Model Building
  + Training the model in multiple algorithms
  + Testing the model
* Performance Testing & Hyperparameter Tuning
  + Testing model with multiple evaluation metrics
  + Comparing model accuracy before & after applying hyperparameter tuning
* Model Deployment
  + Save the best model
  + Integrate with Web Framework
* Project Demonstration & Documentation
  + Record explanation Video for project end to end solution
  + Project Documentation-Step by step project development procedure

**Prior Knowledge:**

You must have prior knowledge of following topics to complete this project.

* ML Concepts
  + Supervised learning: <https://www.javatpoint.com/supervised-machine-learning>
  + Unsupervised learning: <https://www.javatpoint.com/unsupervised-machine-learning>

### Relevant Algorithms (examples, specific algorithms depend on approach - classification/regression)

* + Decision tree:

<https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>

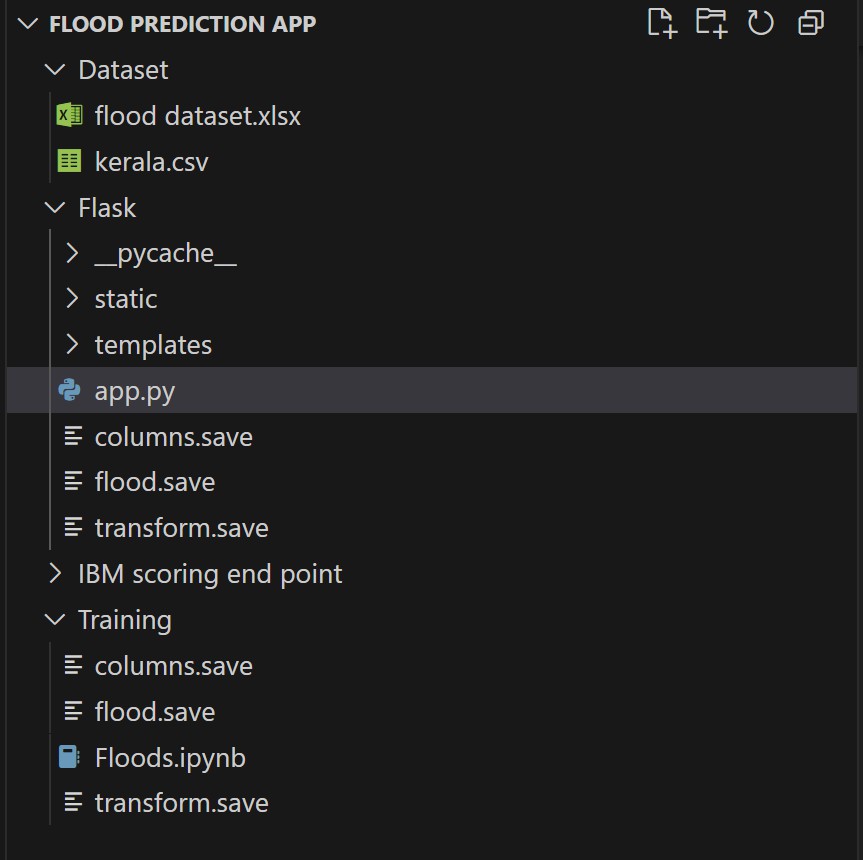
* + Random forest: <https://www.javatpoint.com/machine-learning-random-forest-algorithm>
  + Regression Models (if predicting water level): [Relevant Link - e.g., [https://scikit-](https://scikit-learn.org/stable/supervised_learning.html#regression) [learn.org/stable/supervised\_learning.html#regression](https://scikit-learn.org/stable/supervised_learning.html#regression)]
* Evaluation metrics:

[https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-](https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/) [metrics/](https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/)

* Flask Basics : <https://www.youtube.com/watch?v=lj4I_CvBnt0>

Project Structure:

Create the Project folder which contains files as shown below



* We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
* flood.save is our saved model (name may vary). Further we will use this model for flask integration.
* Data Folder contains the Dataset used
* The Notebook file contains procedures for building the model.

# Milestone 1: Define Problem / Problem Understanding

### Activity 1: Specify the business problem

Floods cause massive damage every year, but existing prediction systems are slow, generic, and inaccessible to vulnerable communities. There's a critical need for a real-time, affordable, and hyperlocal flood prediction tool that empowers people to act before disaster strikes.

### Activity 2: Business requirements

A flood prediction project can have a variety of business requirements, depending on the specific goals and objectives (e.g., predicting flood occurrence, water level, or timing). Some potential requirements may include:

* ∙ Accurate and timely predictions: The system should provide accurate predictions with sufficient lead time to allow for preparation and response.
* ∙ Integration of diverse data sources: The system should be able to integrate data from various sources such as weather stations, river gauges, satellite imagery, and topographical maps.
* ∙ Spatial resolution: The predictions should be specific to geographic areas or points of interest.
* ∙ Robustness: The model should perform reliably under varying environmental conditions.
* ∙ User-friendly interface: The prediction output should be easy to access and understand for relevant stakeholders (e.g., disaster management teams, public).

## Activity 3: Literature Survey

A literature survey for a flood prediction project would involve researching and reviewing existing studies, articles, and other publications on the topic of flood modeling and prediction using machine learning. The survey would aim to gather information on current approaches, different machine learning techniques applied, data sources used, their strengths and weaknesses, and any gaps in knowledge that the project could address. The literature survey would also look at the methods and techniques used in previous flood prediction projects, and any relevant data or findings that could inform the design and implementation of the current project.

**Activity 4: Social or Business Impact.**

Social Impact :- Reduced casualties and displacement: By providing accurate and timely warnings, the system can help save lives and minimize the number of people affected by floods. Reduced property damage: Early warnings allow people to take protective measures for their property.

Business Model/Impact :- Improved disaster management: Enables authorities to deploy resources effectively and plan evacuation routes. Informs urban planning and infrastructure development: Helps identify high-risk areas and design resilient structures. Potential for commercial applications in risk assessment for insurance or real estate.

# Milestone 2: Data Collection & Preparation

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download or access the required dataset.

### Activity 1: Collect the dataset

There are many potential sources for collecting flood-related data, including meteorological agencies (rainfall, temperature), hydrological agencies (river levels, flow rates), topographical data sources (elevation, slope), soil data, and historical flood records.

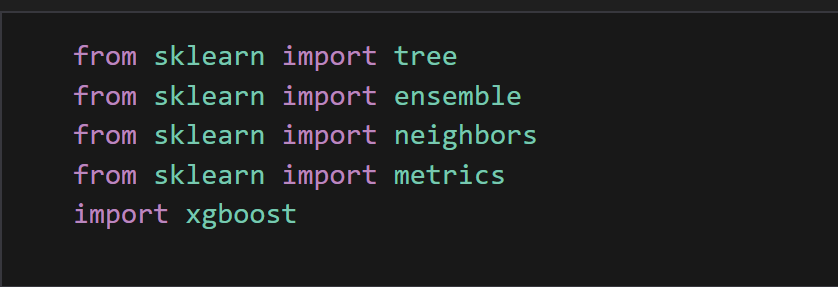
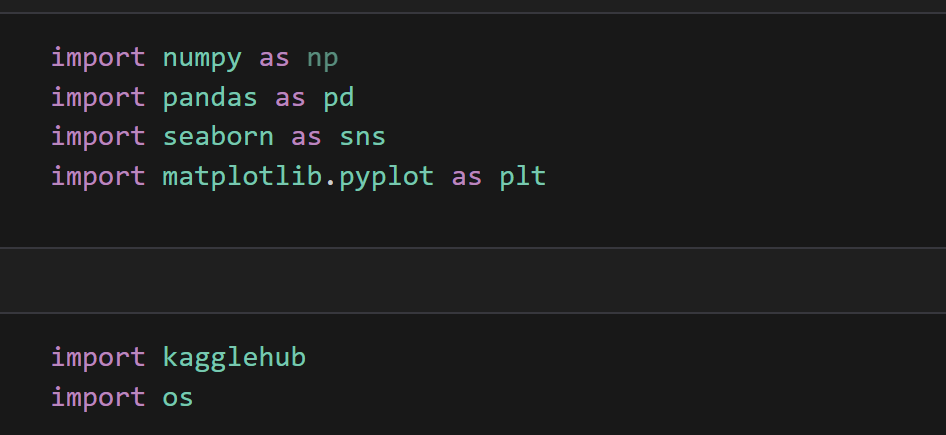
In this project we could use .csv data compiled from various sources. Please refer to the link given below to download or access the dataset.

Link1: <https://www.kaggle.com/datasets/arbethi/rainfall-dataset> Link2: <https://www.kaggle.com/datasets/mukulthakur177/kerela-flood>

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

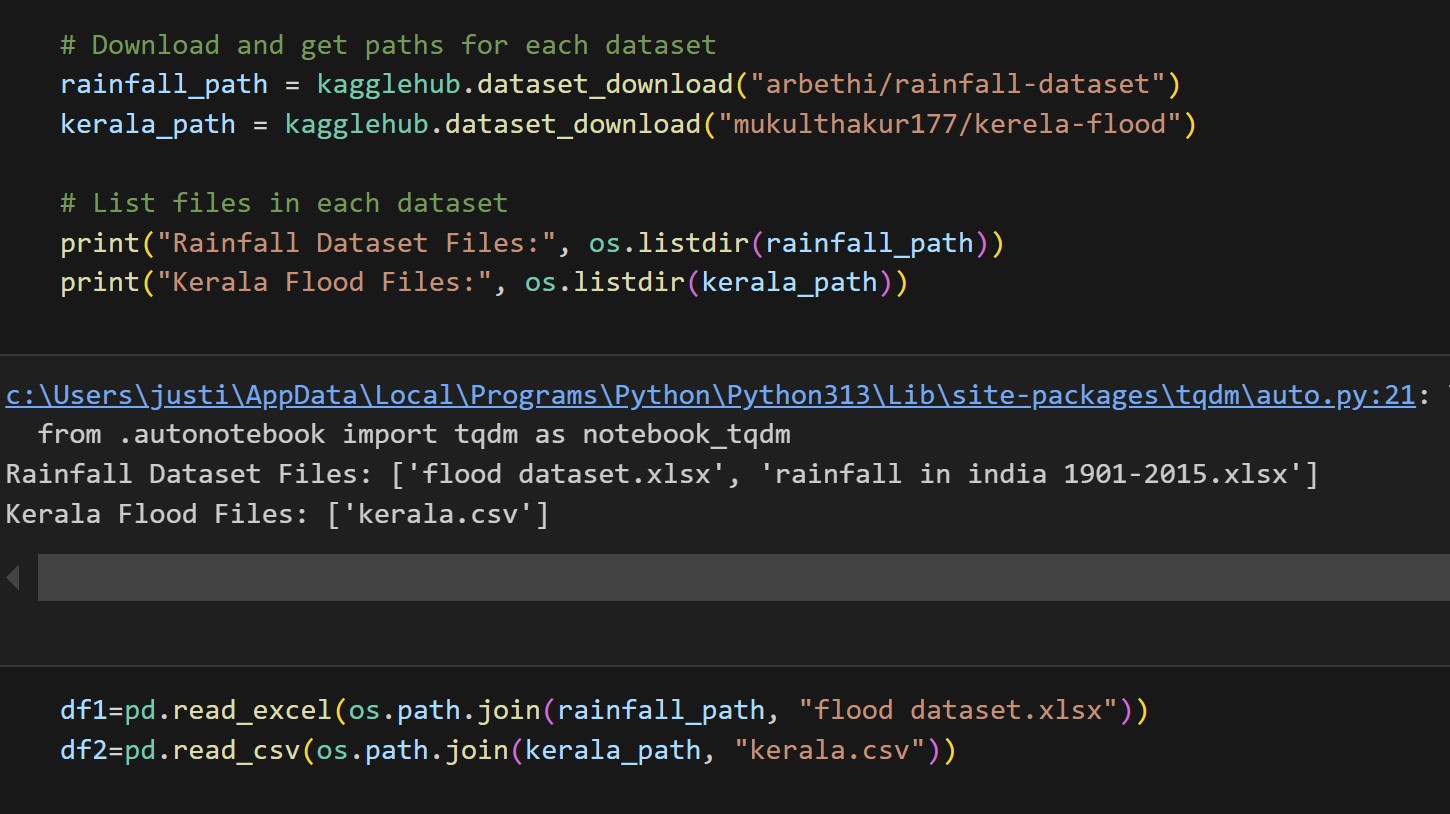
*Note*: There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

##### Activity 1.1: Importing the libraries

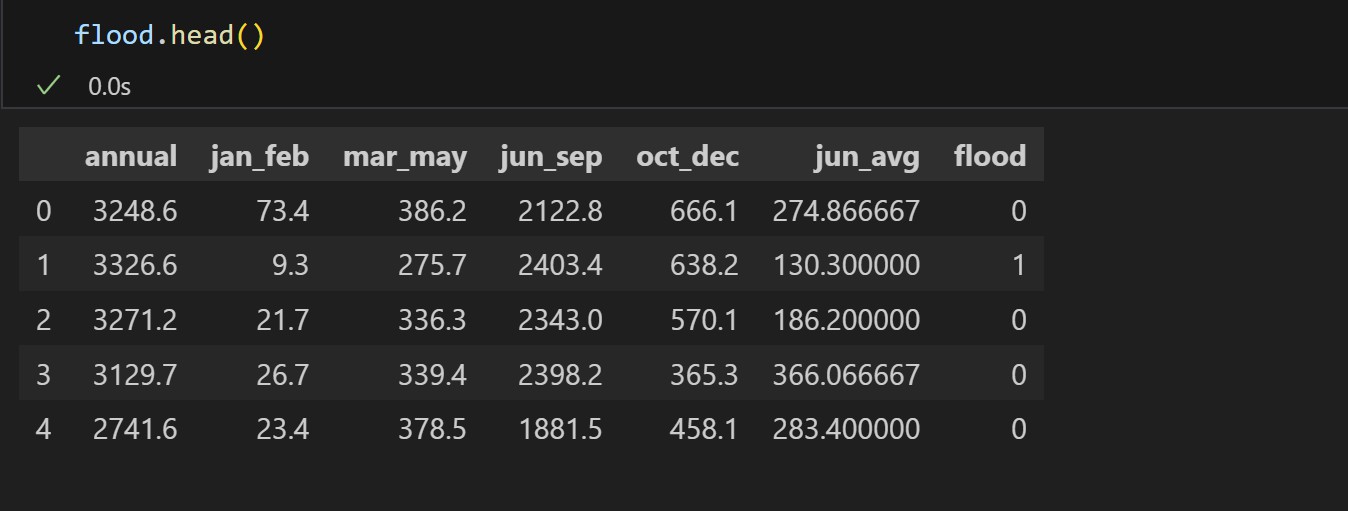
Import the necessary libraries as shown in the image. (optional) Here we have used visualisation style as fivethirtyeight.

##### Activity 1.2: Read the Dataset

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of the csv file.

The dataframes are merged to created flood dataframe



# Activity 2: Data Preparation

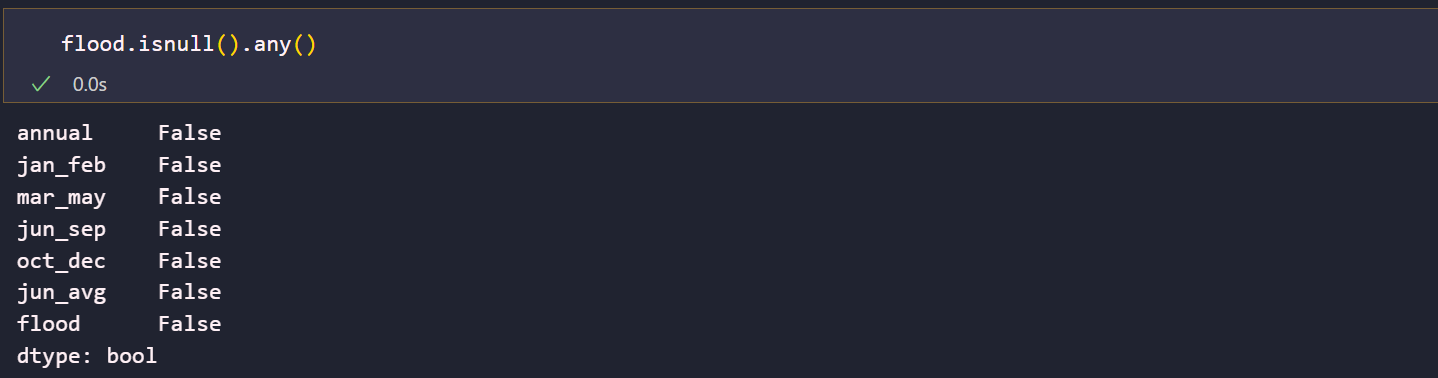
As we have understood how the data is, let's pre-process the collected data.

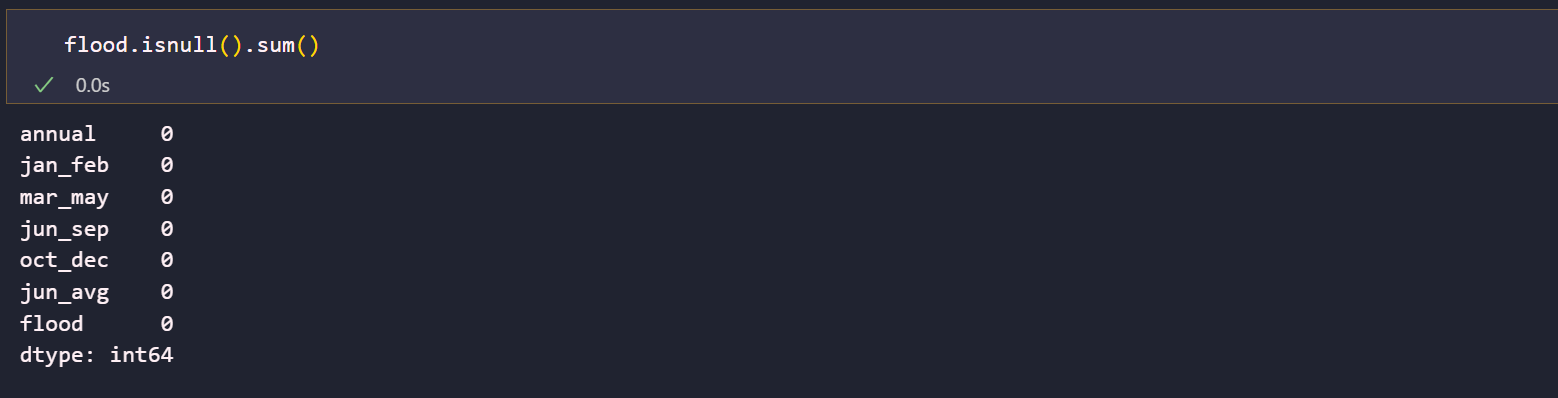
The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

* Handling missing values
* Handling Outliers
* Feature Engineering (Creating features relevant to flood prediction like cumulative rainfall, rate of change of water level, time since last rainfall event, etc.)

*Note:* These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

#### Activity 2.1: Handling missing values

* + For checking the null values, df.isnull().any( ) function is used. To sum those null values we use df.isnull().sum() function. From the below image we found that there are/are no null values present in our dataset. If there are null values, imputation or removal strategies are needed.
* [Repeat/Clarify checking nulls description if needed]



## Milestone 3: Exploratory Data Analysis

#### Activity 1: Descriptive statistical

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, mi, max and percentile values of continuous features (e.g., rainfall, temperature, river level).

### Activity 2: Visual analysis

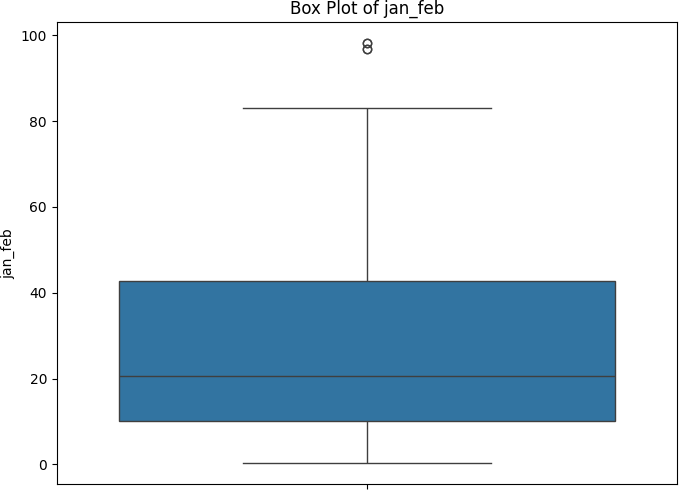
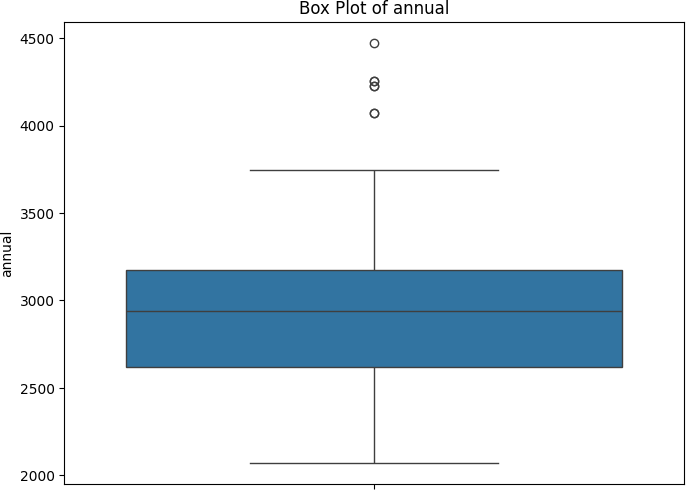
Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions relevant to flood prediction (e.g., seeing rainfall patterns, seasonal trends (june-sept in our case) in river levels, correlation between rainfall and flood events).

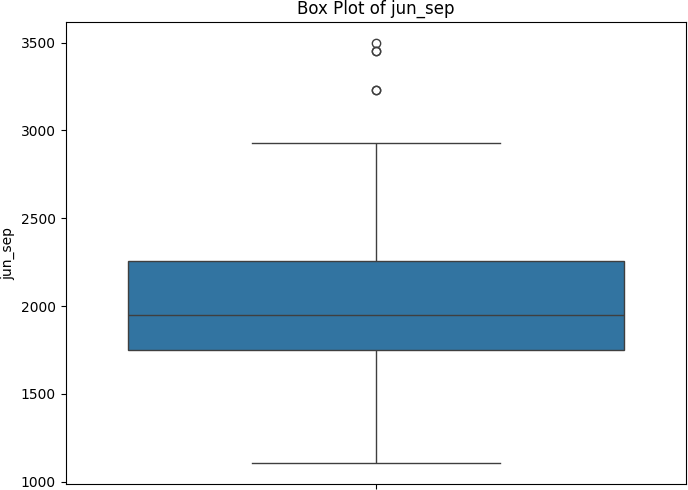
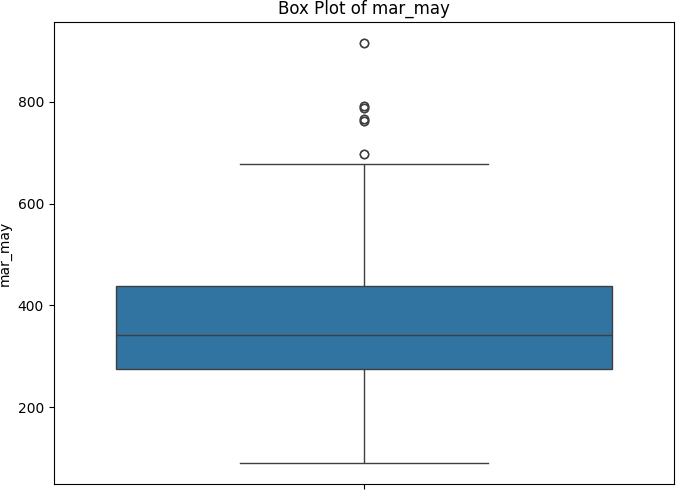
#### Activity 2.1: Univariate analysis

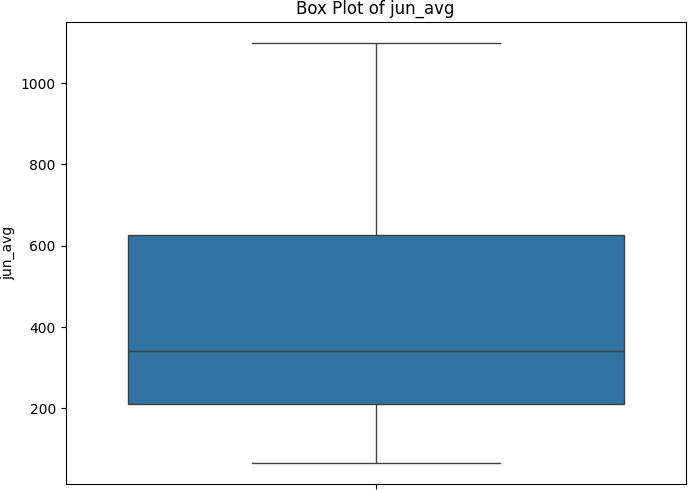
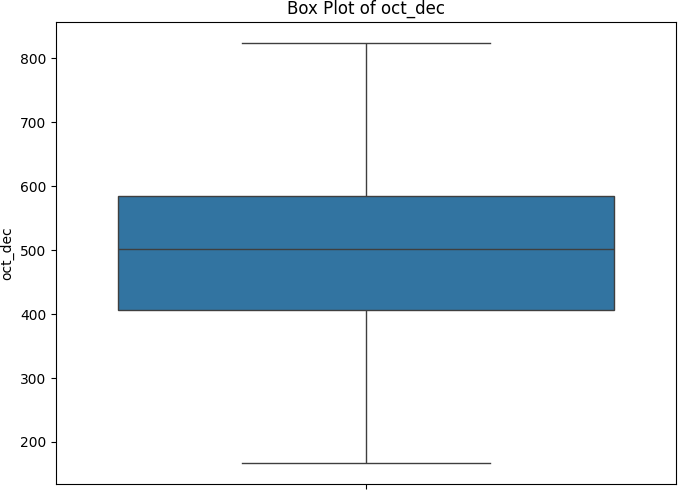
In simple words, univariate analysis is understanding the data with single feature. Here we have displayed different graphs such as countplot for categorical features (e.g., location, land use) and histograms for continuous features (e.g., rainfall, river level).

Seaborn package provides a wonderful function countplot. It is more useful for

categorical features. With the help of countplot, we can see the frequency of unique values. For histograms, we can see the distribution of values for continuous features.



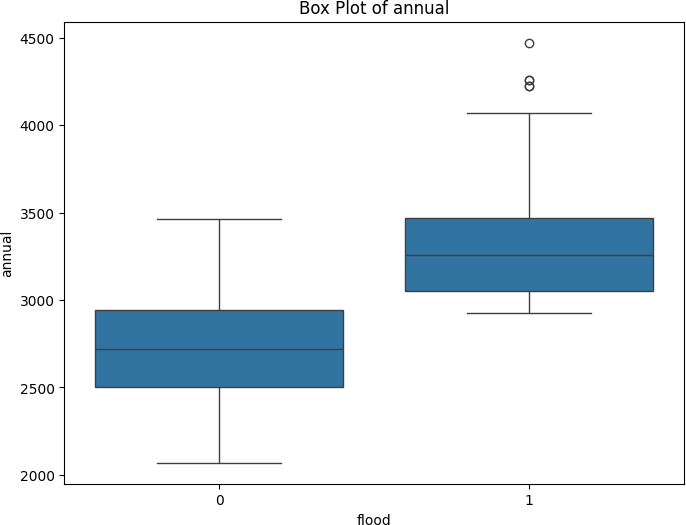
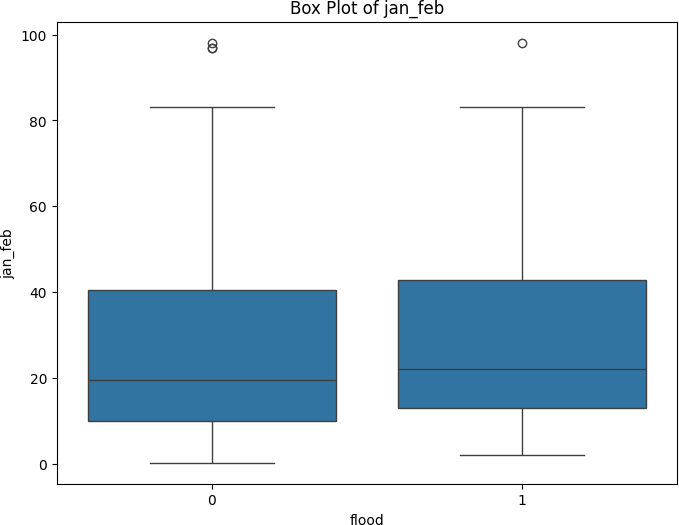


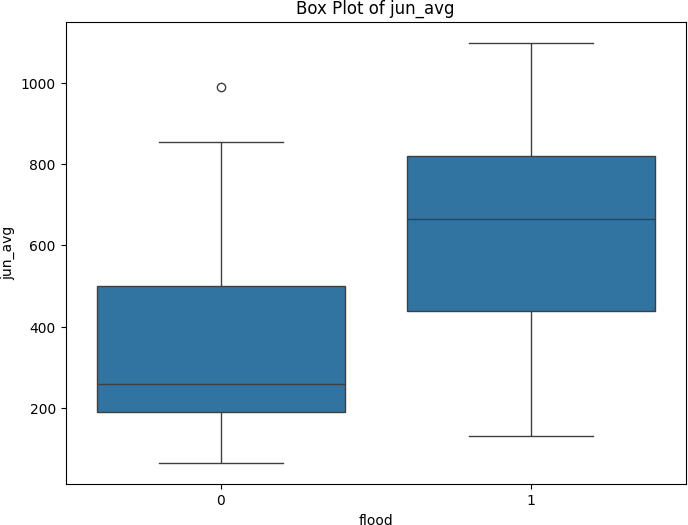
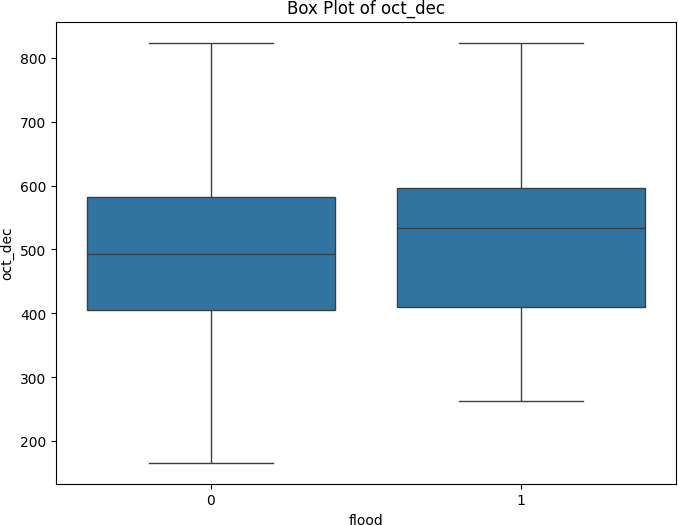


#### Activity 2.2: Bivariate analysis

To find the relation between two features we use bivariate analysis. Here we can use barplot, scatter plot, or line plots (especially for time series data).

* Barplot or scatter plot is used here. As parameters, we are passing two features, e.g., 'rainfall' and 'river\_level'.



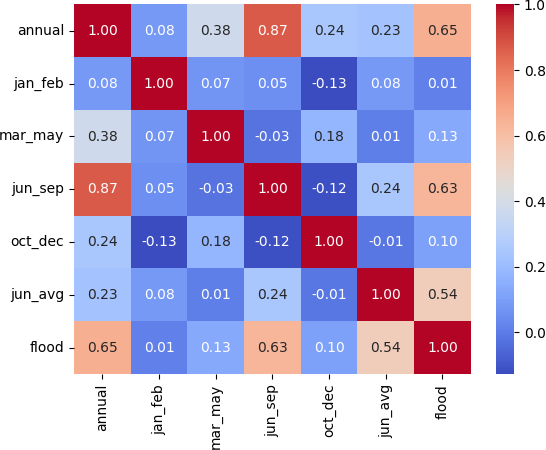
* From the below plot you can understand the relationship between the two features.
* [Description of a specific bivariate plot, e.g., scatter plot of rainfall vs. river level, line plot showing river level over time

#### Activity 2.3: Multivariate analysis

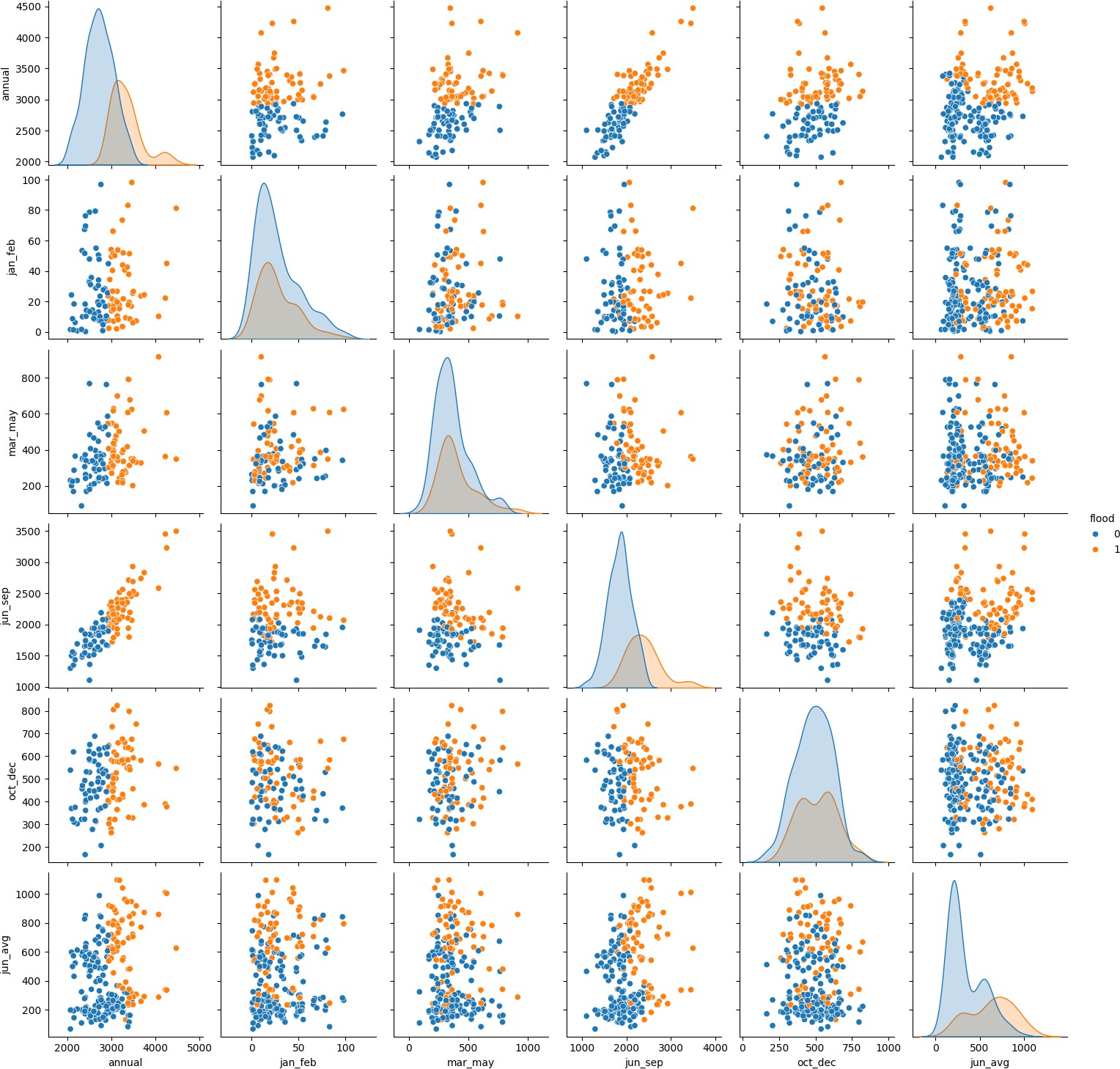
In simple words, multivariate analysis is to find the relation between multiple features.

Here we have used heatmap from seaborn package or other methods like pair plots.

* From the below image (heatmap), we came to a conclusion that there are some features which are highly correlated.

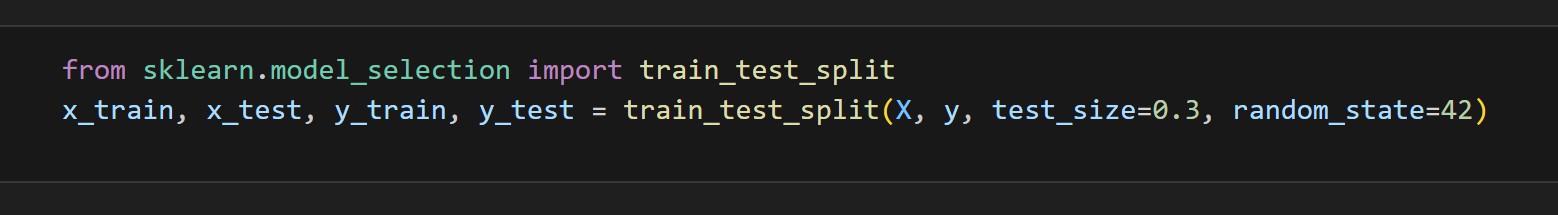


* + ∙ The correlation between different environmental factors can be observed. Features strongly correlated with the target variable (flood occurrence/level) are important.
  + ∙ Features that are highly correlated with each other (multicollinearity) might need to be handled (e.g., dropping one).



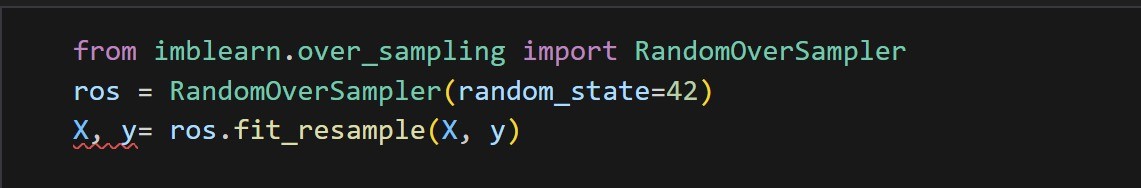
**Splitting data into train and test**

Now let’s split the Dataset into train and test sets. First split the dataset into x (features) and y (target variable - flood/no flood, or water level) and then split the data set.

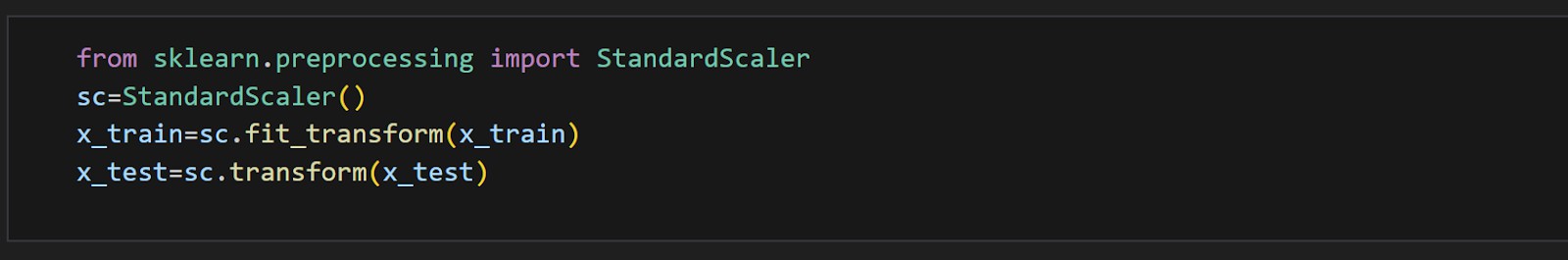
Here x and y variables are created. On x variable, df is passed with dropping the target variable. And on y target variable is passed. For splitting training and testing data we are using train\_test\_split() function from sklearn. As parameters, we are passing x, y, test\_size, random\_state.

**Handling Imbalanced dataset**

* Flood events are often rare compared to normal conditions, leading to an imbalanced dataset. Handling imbalanced data is important to ensure that the model is not biased towards the majority class (no flood) and can accurately predict the minority class (flood).
* Here we are using techniques like SMOTE, Oversampling, or Undersampling.



## Feature Scaling

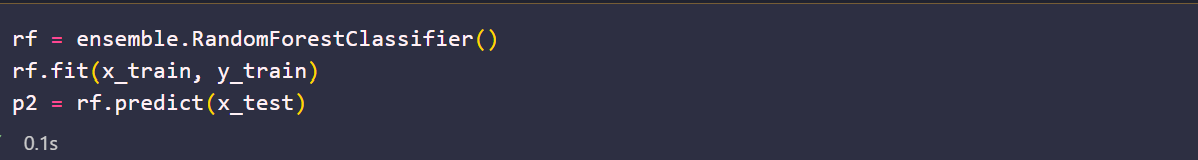
* Scaling is a technique used to transform the values of a dataset to a similar scale to improve the performance of machine learning algorithms that are sensitive to feature scales (e.g., SVM, KNN, models using gradient descent).
* Here we are using Standard Scaler or Min-Max Scaler.
* This scales the data according to the chosen method. The formula for Standard Scaling is given by: X\_scaled = (X - X\_mean) / X\_std

# Milestone 4: Model Building

#### Activity 1: Training the model in multiple algorithms

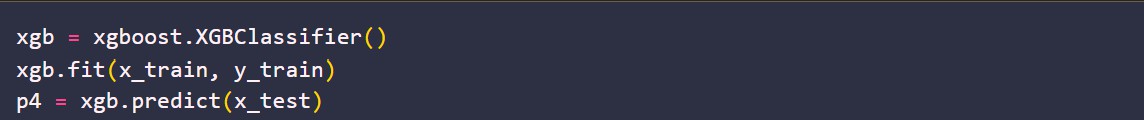
Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms suitable for classification (predicting flood/no flood) or regression (predicting water level). The best model is saved based on its performance.

##### Activity 1.1: [Algorithm 1, e.g., Random Forest] model

First [Algorithm 1] is imported from sklearn Library then [Algorithm 1] algorithm is initialised and training data is passed to the model with the .fit() function. Test data is predicted with .predict() function and saved in a new variable. We can find the Train and Test accuracy/score by X\_train and X\_test.

#### Activity 1.2: [Algorithm 2, e.g., Gradient Boosting or LSTM] model

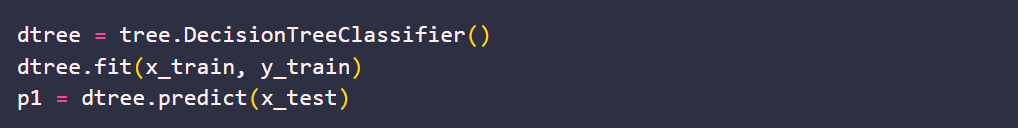
First [Algorithm 2] Model is imported from sklearn Library then [Algorithm 2] algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with

.predict() function and saved in a new variable. We can find the Train and Test accuracy/score by X\_train and X\_test.

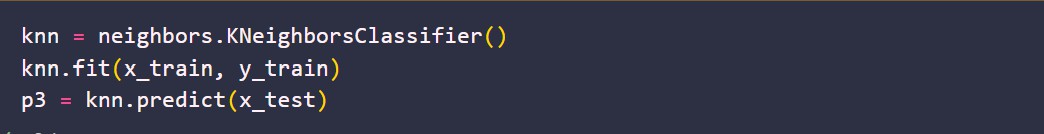
#### Activity 1.3: [Algorithm 3, e.g., Support Vector Machine or Logistic Regression or Decision Tree] model

[Algorithm 3] Model is imported from sklearn Library then [Algorithm 3] algorithm is

initialised and training data is passed to the model with .fit() function. Test data is predicted with

.predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report (for classification) or relevant regression metrics are calculated.

#### Activity 1.4: [Algorithm 4, KNN model]

KNN Model is imported from sklearn Library then KNN algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, relevant metrics are calculated.

### Activity 2: Testing the model

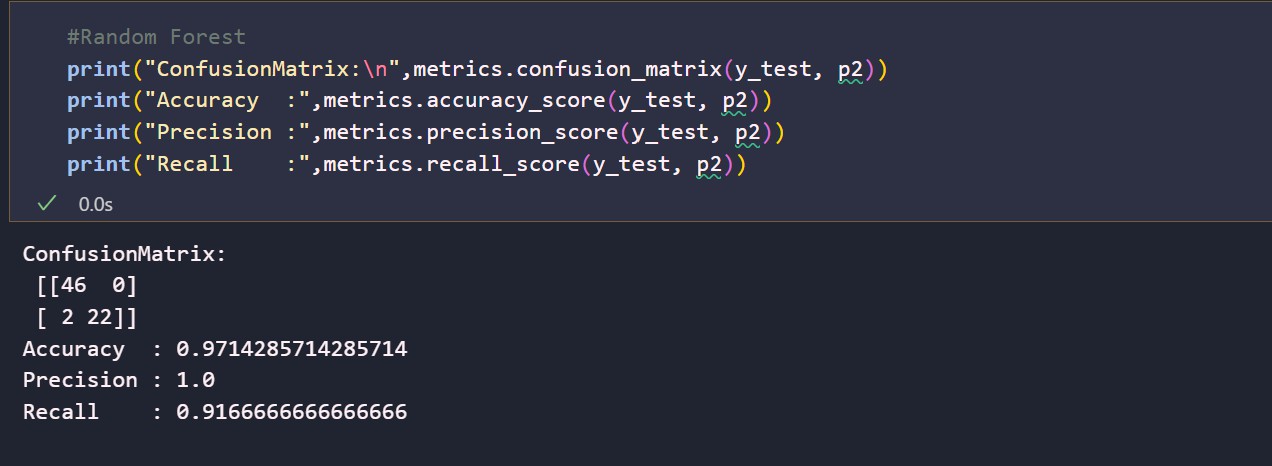
Here we have tested with one or more algorithms. You can test with all algorithms considered. With the help of predict() function, make predictions on the test set.

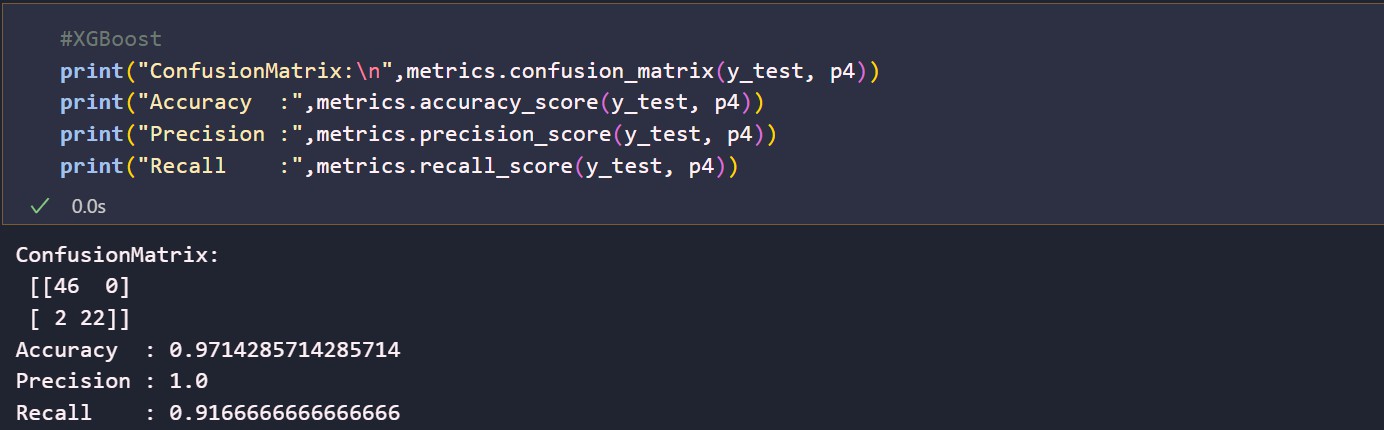
# Milestone 5: Performance Testing & Hyperparameter Tuning

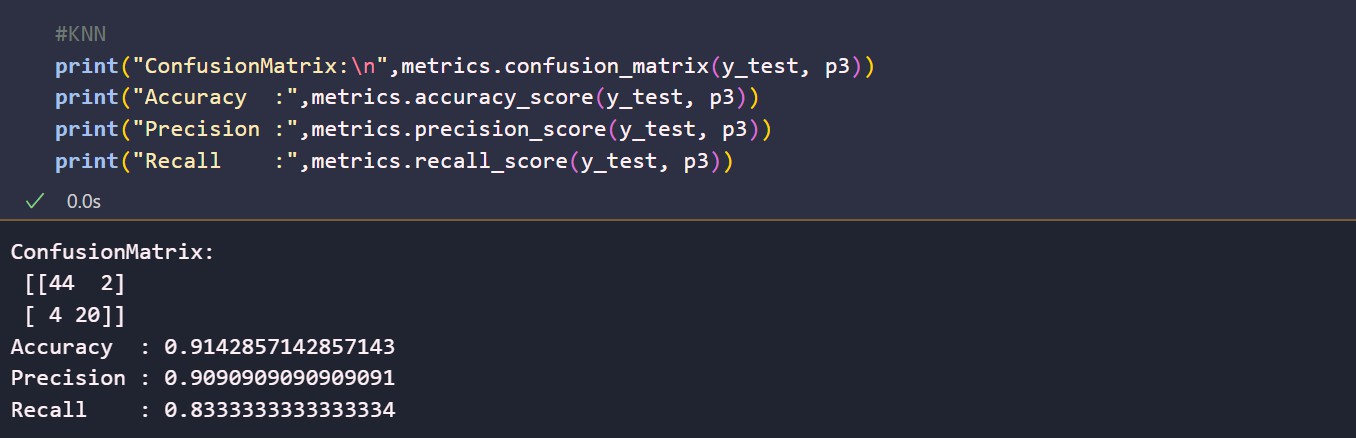
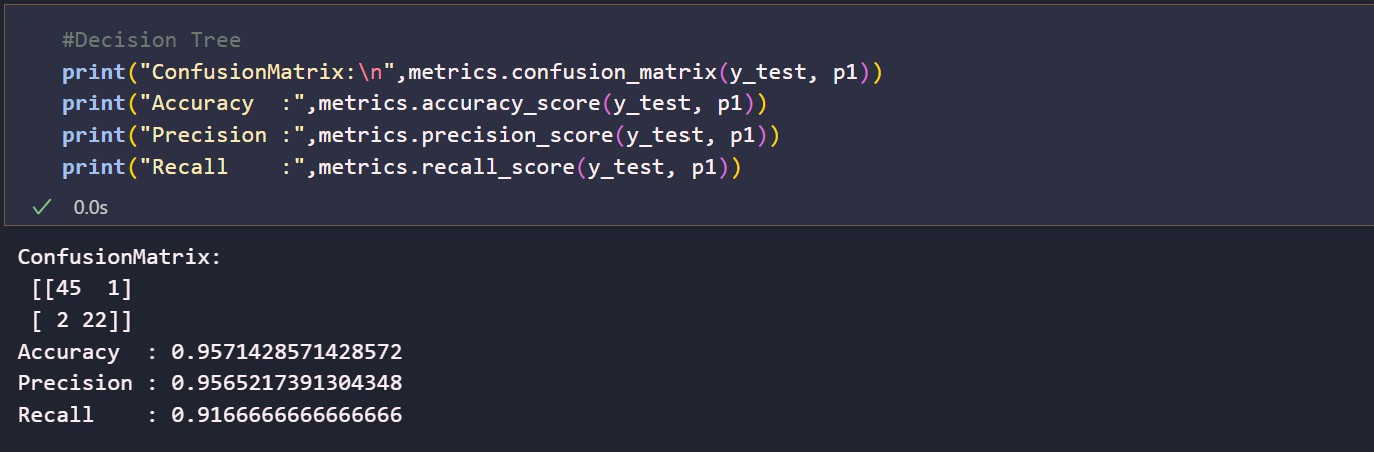
### Activity 1: Testing model with multiple evaluation metrics

Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This is especially important for imbalanced datasets like flood prediction. For classification, metrics like Precision, Recall (very important for not missing flood events), F1-score, and AUC are crucial. For regression (predicting water level), metrics like RMSE, MAE, R-squared are used. We are using evaluation metrics for the specific task (classification or regression).

#### Activity 1.1: Compare the model

For comparing the chosen models, a comparison table or plot showing the performance metrics is generated.





After calling the function or displaying the results, the performance of models are displayed as output. From the above results, [Best Model Name] is performing well based on the chosen key metrics (e.g., Recall for flood prediction).

### Activity 2: Comparing model accuracy before & after applying hyperparameter tuning

Hyperparameter tuning involves optimizing the model's internal parameters to improve

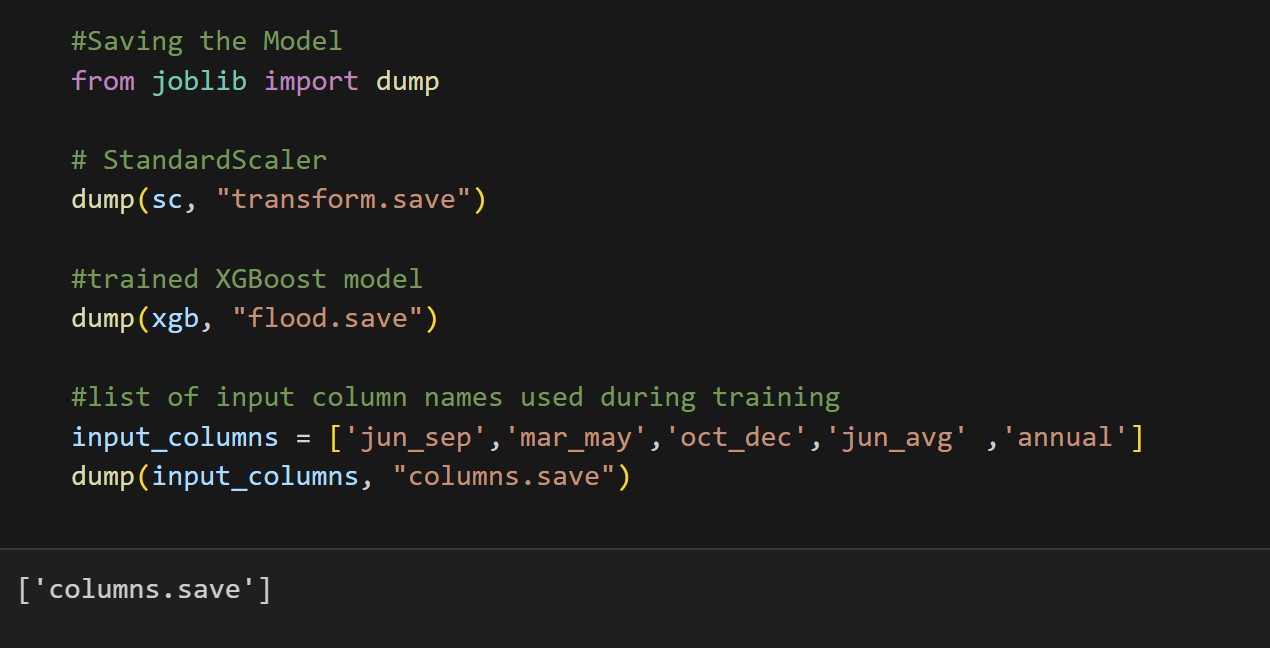
performance. Cross-validation is often used during tuning to get a robust estimate of performance. Evaluating performance of the model using cross\_val\_score from sklearn can

show average performance across different data folds. On the parameters, we have given the model name, x, y, cv (e.g., as 5 folds). Evaluate if tuning improves the model's performance. *Note: To understand cross validation, refer to this link*

Link: <https://www.geeksforgeeks.org/machine-learning/cross-validation-machine-learning/>

# Milestone 6: Model Deployment

### Activity 1: Save the best model

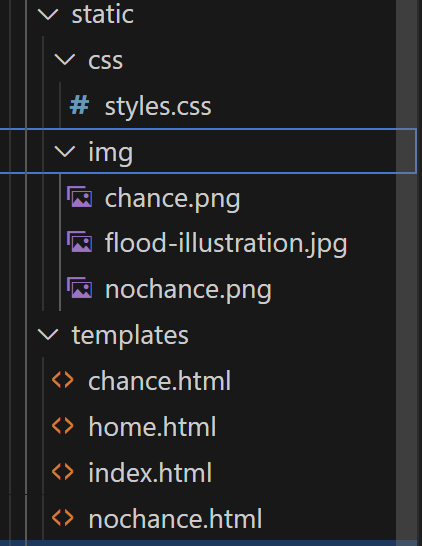
Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance based on project goals. This is useful for future use without retraining. Using libraries like `pickle` or `joblib` to save the model object.

### Activity 2: Integrate with Web Framework

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the users (or system operators) where they can enter input values relevant to prediction (e.g., current rainfall, river level, forecast data). The entered values are given to the saved model and the prediction (e.g., "No Flood Expected", "Flood Alert Level 1", Predicted Water Level) is showcased on the UI.

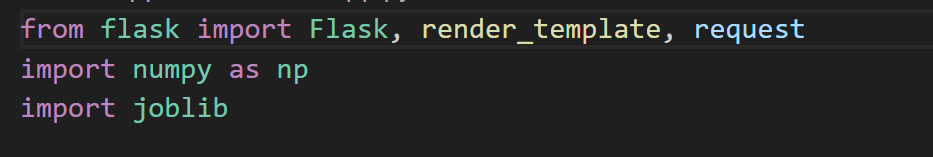
This section has the following tasks

* Building HTML Pages
* Building server-side script
* Run the web application **Activity 2.1: Building Html Page:** For this project create HTML file namely
* index.html (for input)
* predict.html (for displaying results) and save them in the templates folder.



### Activity 2.2: Build Python code:

Import the libraries (Flask, libraries needed to load and use the model)

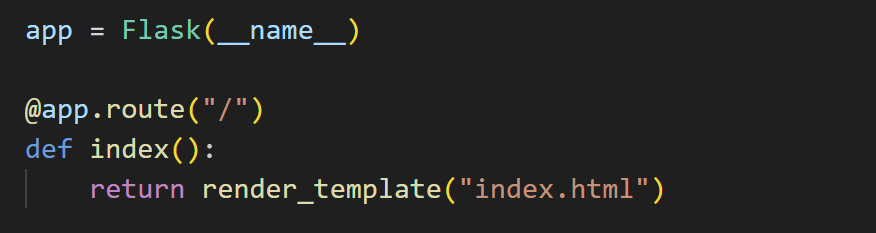


Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module ( name ) as argument.



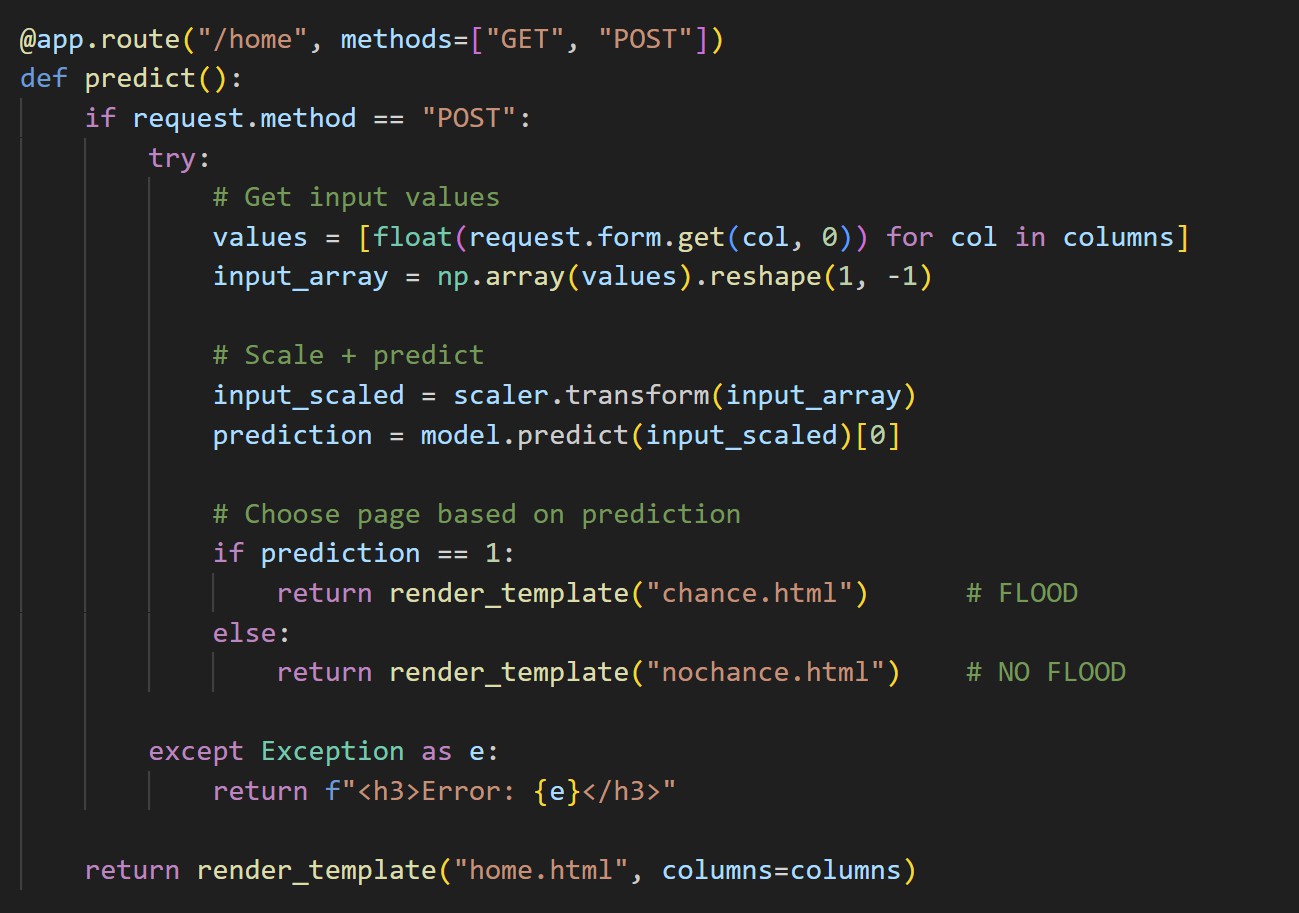
**Render HTML page:**

Here we will be using a declared constructor to route to the HTML page which we have created earlier.

In the above example, ‘/’ URL is bound with the index.html function, which renders the input form.

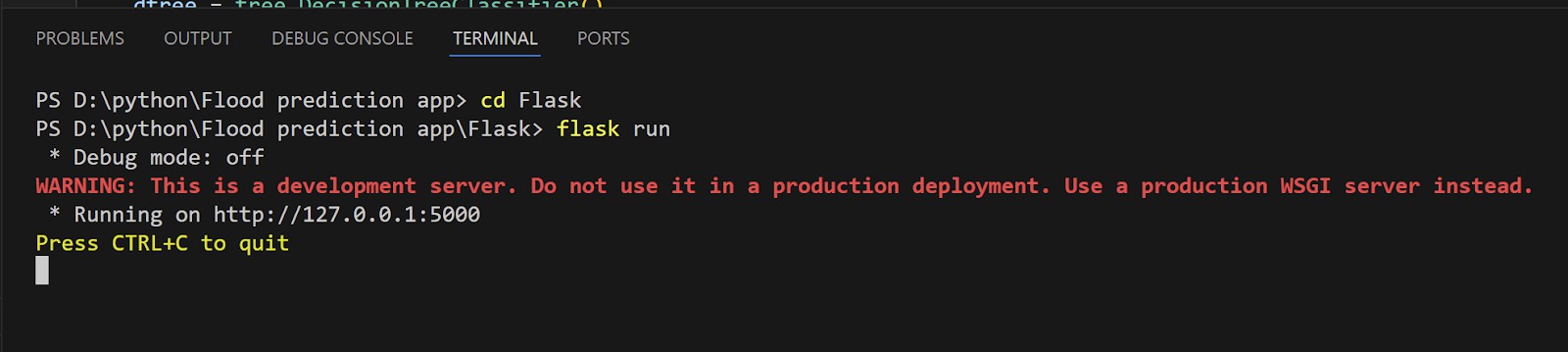
Whenever you enter the values from the html page the values can be retrieved using POST Method.

**Retrieves the value from UI:**

Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. Those values are processed and formatted into the input format expected by the trained model. This formatted input is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the predict.html page earlier.

**Main Function:**

Standard Flask app run command.

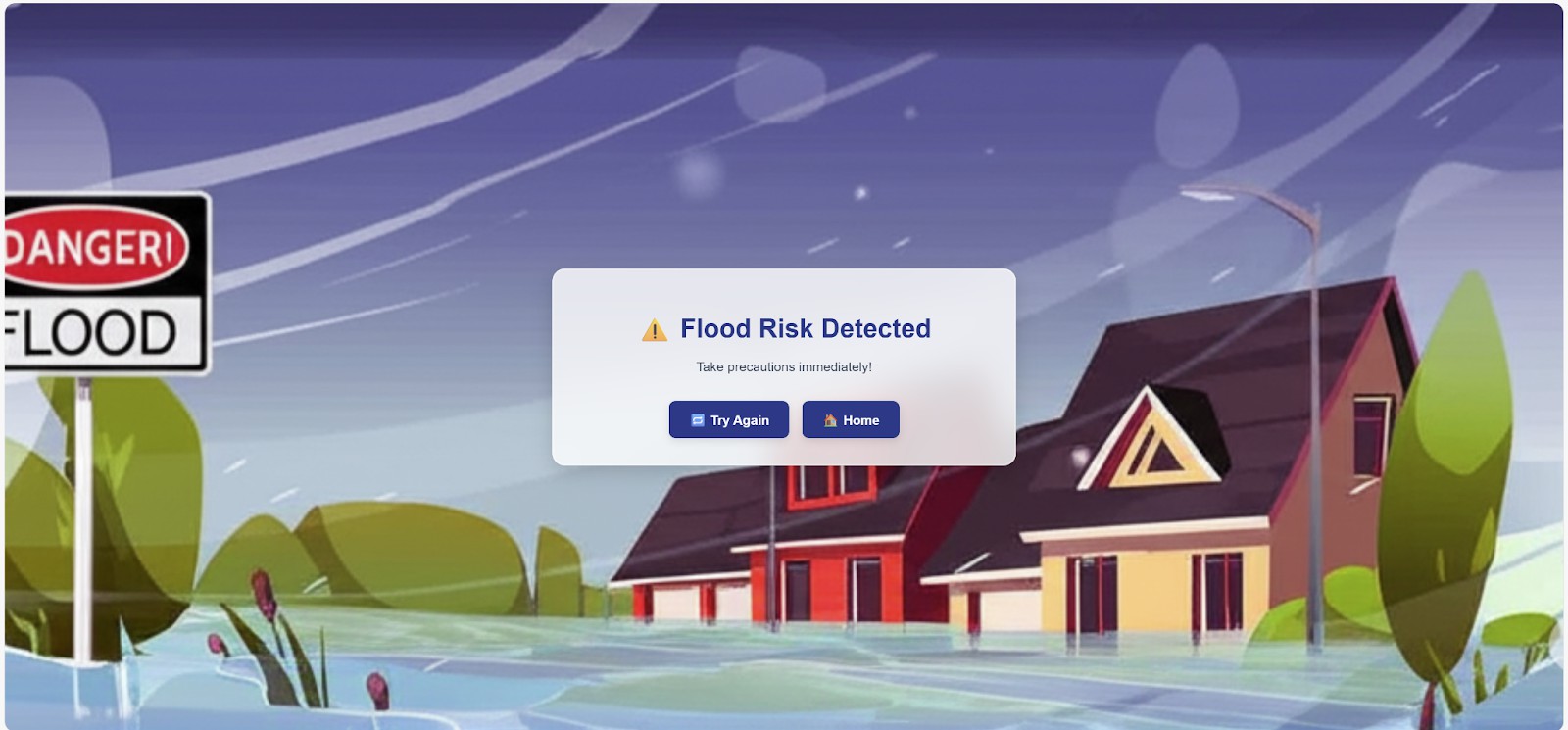
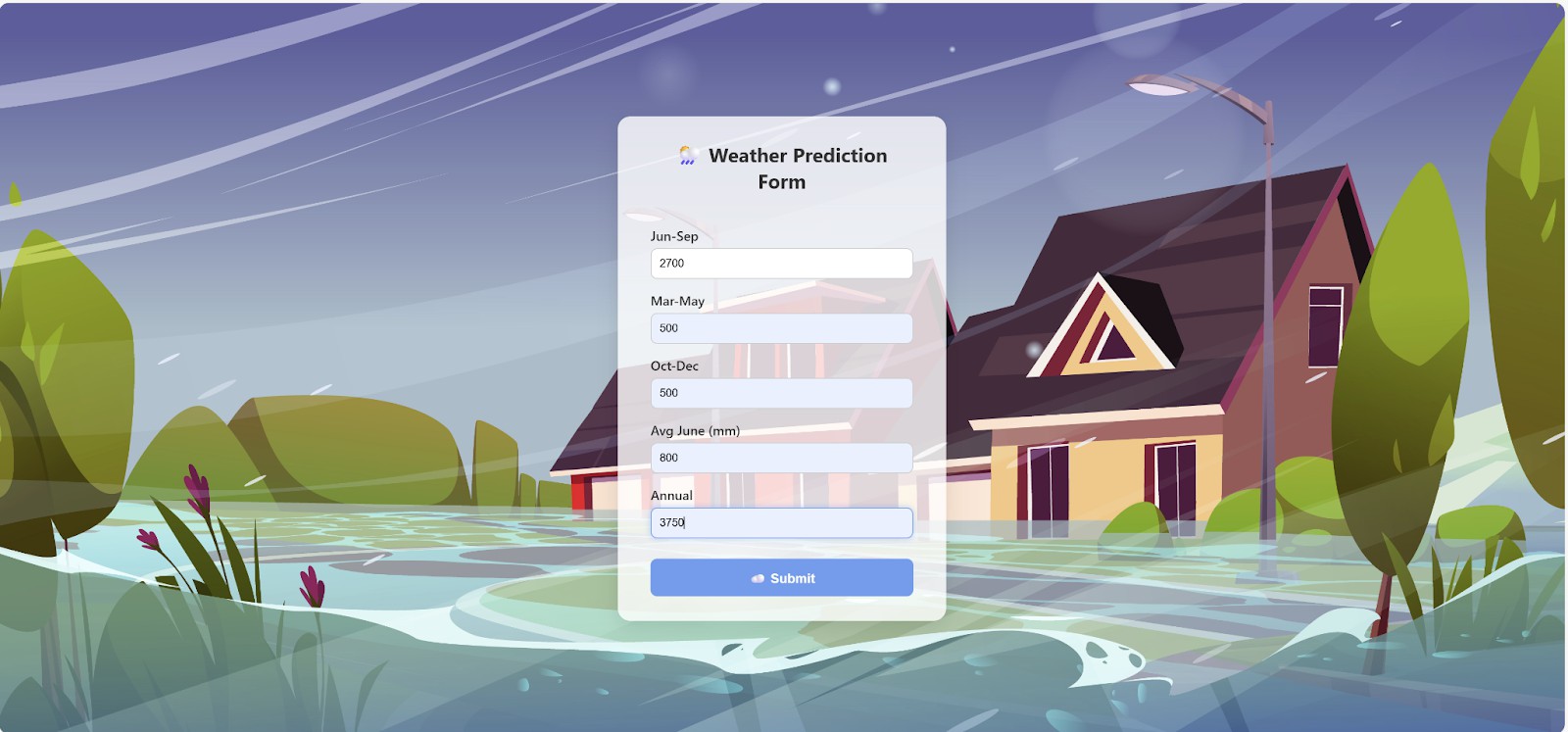


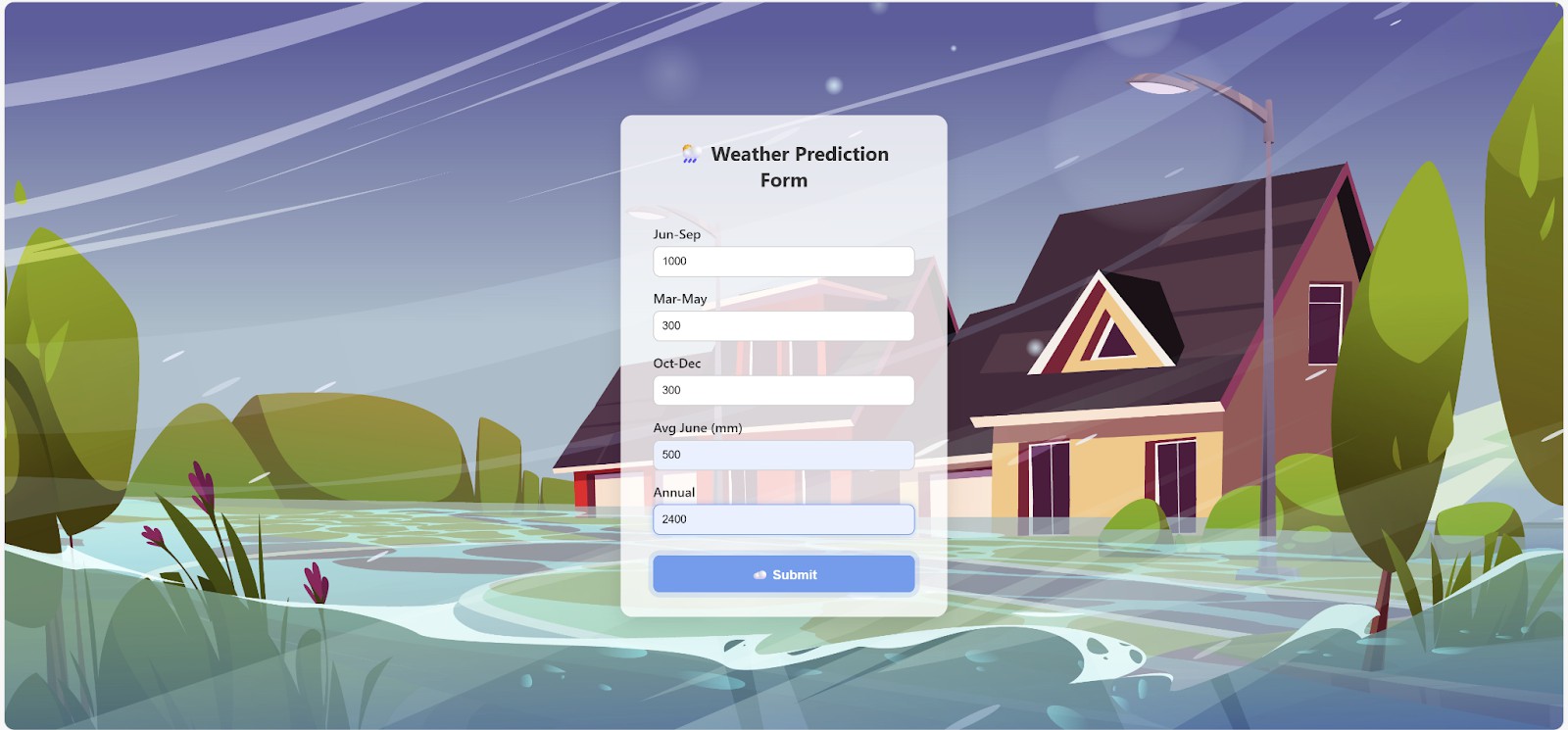
### Activity 2.3: Run the web application

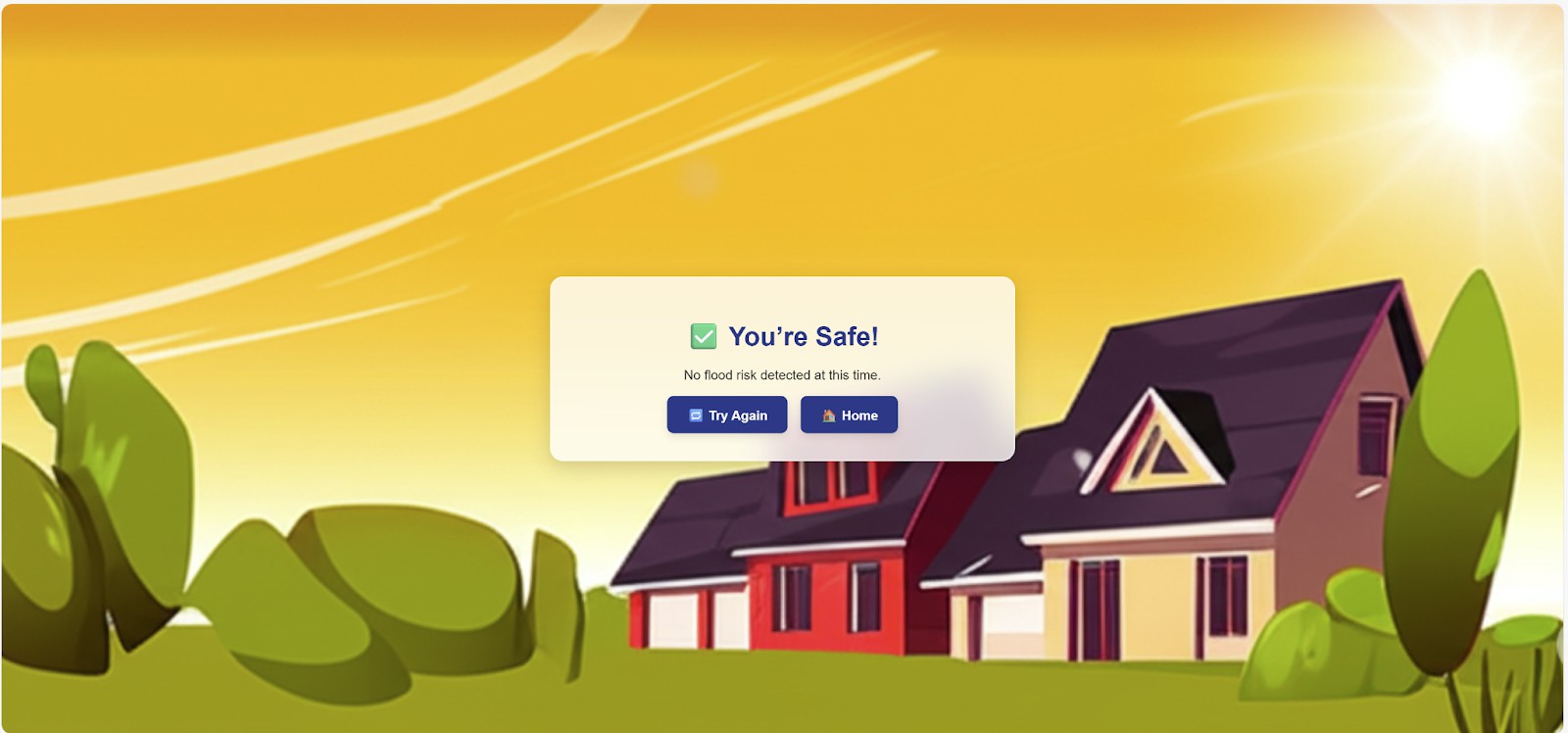
* Open command prompt or terminal.
* Navigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page (e.g., [http://127.0.0.1:5000).](http://127.0.0.1:5000/)
* Enter the required input values (e.g., rainfall, river levels), click on the submit button, and see the result/prediction on the web page.

Now,Go the web browser and write the localhost url ([http://127.0.0.1:5000)](http://127.0.0.1:5000/) to get the below result (input page).

Sample input and Output







# Milestone 7: Project Demonstration & Documentation

Below mentioned deliverables to be submitted along with other deliverables

**Activity 1**:- Record explanation Video for project end to end solution

**Activity 2**:- Project Documentation-Step by step project development procedure Create document as per the template provided