**Project Title: Development of a Machine Learning Model for Rain Prediction in the UK’s Highest Rainfall County**

**[Name of the Student]**

**[Name of the Institution]**

**Table of Contents**

[**1. Data Collection and Preparation** 3](#_Toc173198276)

[Dataset 3](#_Toc173198277)

[Data Cleaning 3](#_Toc173198278)

[Exploring the data 3](#_Toc173198279)

[Insightful Visualizations of the Existing Dataset 5](#_Toc173198280)

[Feature Engineering 8](#_Toc173198281)

[1 .Time-Based Features 8](#_Toc173198282)

[2. Temperature Interactions with rainfall 9](#_Toc173198283)

[3. Rolling and Cumulative Features 10](#_Toc173198284)

[4. Creating Lag Features (Lag Effects) 11](#_Toc173198285)

[5. Rain Analysis with Humidity and Temperature 12](#_Toc173198286)

[6. Analyzing Wind Speed with Atmospheric Pressure on Rain 13](#_Toc173198287)

[7. Climate Indices 13](#_Toc173198288)

[8. Rainfall Intensity (amount of rain per hour) 15](#_Toc173198289)

[9. Topographical Features (Data Elevation) 16](#_Toc173198290)

[10. Wind Direction 17](#_Toc173198291)

[Important Feature Selecting, Dropping Irrelevant Features 19](#_Toc173198292)

[**2. Model Development** 19](#_Toc173198293)

[Algorithm Selection 19](#_Toc173198294)

[Dataset Splitting 20](#_Toc173198295)

[Model Training 20](#_Toc173198296)

[Linear Regression 20](#_Toc173198297)

[Ridge Regression 20](#_Toc173198298)

[Random Forest Regressor 20](#_Toc173198299)

[Feature Importance 20](#_Toc173198300)

[**3. Model Evaluation** 20](#_Toc173198301)

[Performance Metrics 21](#_Toc173198302)

[Regression Tasks 21](#_Toc173198303)

[Evaluations 21](#_Toc173198304)

[Results 21](#_Toc173198305)

[Selecting the Best Model 21](#_Toc173198306)

[Classification Tasks 22](#_Toc173198307)

[Evaluation Metrics 22](#_Toc173198308)

[Results 22](#_Toc173198309)

[Optimization 23](#_Toc173198310)

[Hyperparameter Tuning 23](#_Toc173198311)

[Evaluate Optimized Model 23](#_Toc173198312)

# 1. Data Collection and Preparation

## Dataset

This project uses a historical weather dataset for Cumbria, which is downloaded from the NASA Power Data Access Viewer. The dataset includes the following features with the following data types:

*Month: Categorical (Object)*

*Year: Integer*

*AP (Atmospheric Pressure): Float*

*RH2M Humidity (This represents the relative humidity at 2 meters above the ground.): Float*

*WS2M Wind Speed (This represents the wind speed at 2 meters above the ground.): Float*

*Temp (Max): Float*

*Temp (Min): Float*

*Past Precipitation Records: Float*

Initially, the dataset contains 276 entries and 8 columns.

### Data Cleaning

***Importing Libraries***

The first step taken in this project is importing the necessary libraries. The following the codes of libraries being utilized in the project for the data-cleaning process:

|  |
| --- |
| import pandas as pd  import numpy as np  import seaborn as sns  import matplotlib.pyplot as plt  from scipy import stats |

***Loading the Dataset***

Another step is loading the dataset from an Excel file named 'data.xlsx':

|  |
| --- |
| import pandas as pd  import numpy as np  import seaborn as sns  import matplotlib.pyplot as plt  from scipy import stats |

### Exploring the data

The data is then explored to understand its structure and find areas to be cleaned. Upon studying the sfeatures, the columns naming “'RH2M Humidity” and “WS2M Wind Speed” are renamed to “Humidy” and “Wind Speed”. Hence, the updated data is:

Table 1 First 6 rows of the data

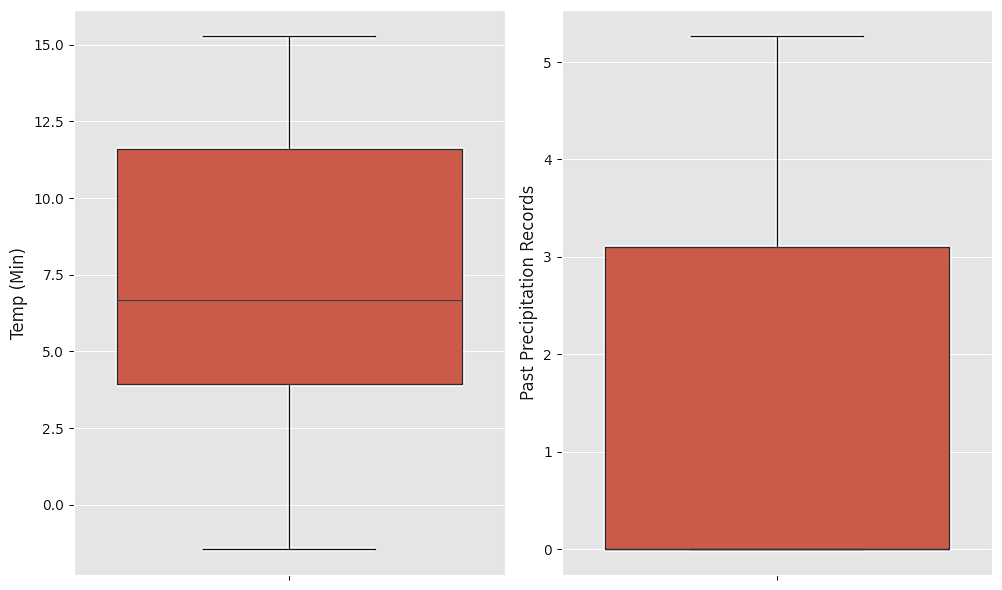
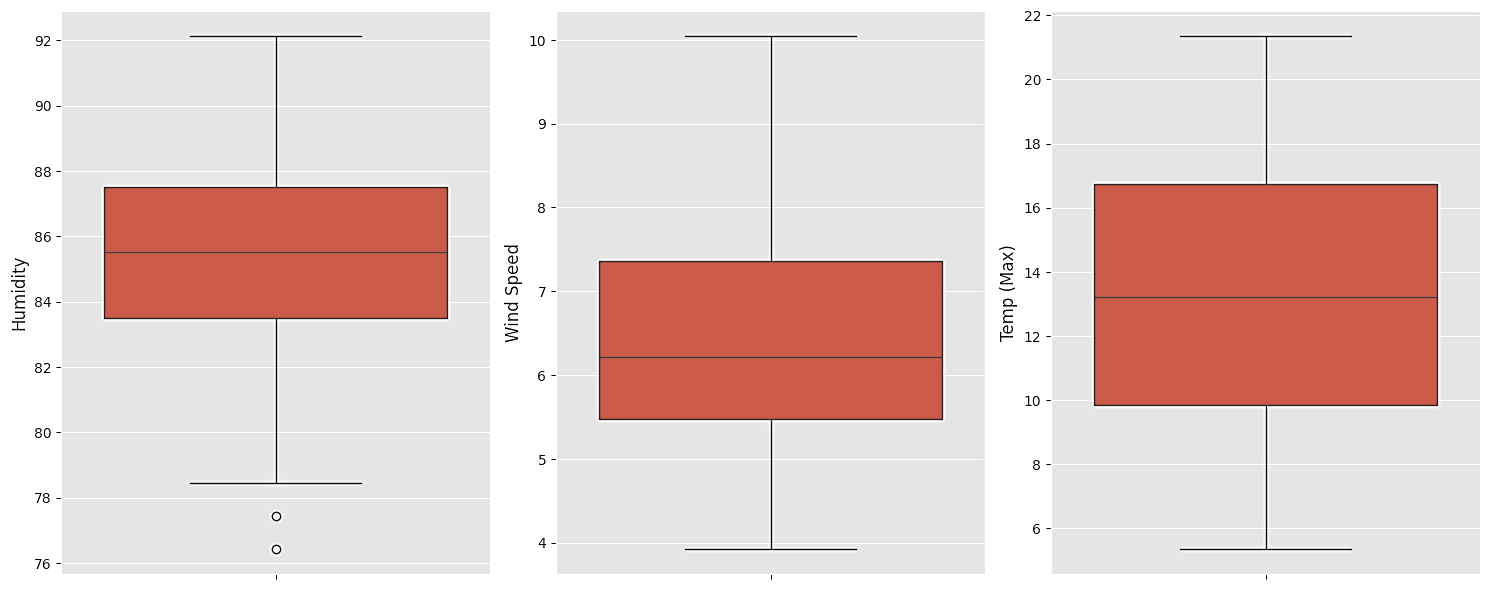
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Month** | **Year** | **AP (Atmospheric Pressure)** | **Humidity** | **Wind Speed** | **Temp (Max)** | **Temp (Min)** | **Past Precipitation Records** |
| Jan | 2022 | 102 | 86.5 | 7.46 | 10.01 | 4.25 | 1.41 |
| Feb | 2022 | 100.76 | 84.75 | 9.6 | 10.16 | 4.3 | 3.75 |
| Mar | 2022 | 102.4 | 86.31 | 5.49 | 9.3 | 3.83 | 1.25 |
| Apr | 2022 | 101.57 | 85.19 | 5.91 | 9.65 | 4.71 | 0.98 |
| May | 2022 | 101.63 | 88.31 | 5.19 | 13.15 | 7.64 | 1.35 |

Table 2 Statistic of the data

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Year** | **AP (Atmospheric Pressure)** | **Humidity** | **Wind Speed** | **Temp (Max)** | **Temp (Min)** | **Past Precipitation Records** |
| **count** | 276 | 276 | 276 | 276 | 276 | 276 | 276 |
| **mean** | 2011 | 101.338 | 85.4264 | 6.45214 | 13.232 | 7.57649 | 1.43007 |
| **std** | 6.6453 | 0.50341 | 2.88408 | 1.26622 | 3.83382 | 4.19552 | 2.23889 |
| **min** | 2000 | 99.55 | 76.44 | 3.93 | 5.37 | -1.45 | 0 |
| **25%** | 2005 | 101.068 | 83.5 | 5.4775 | 9.8575 | 3.925 | 0 |
| **50%** | 2011 | 101.39 | 85.53 | 6.215 | 13.21 | 6.68 | 0 |
| **75%** | 2017 | 101.65 | 87.5 | 7.36 | 16.73 | 11.605 | 3.105 |
| **max** | 2022 | 102.61 | 92.12 | 10.04 | 21.34 | 15.29 | 5.27 |

***Cleaning the Dataset***

No missing values or inconsistencies were found. The box plots are generated for various columns for outlier detection.



**Figure 1 Visualizing outliers with boxplots**

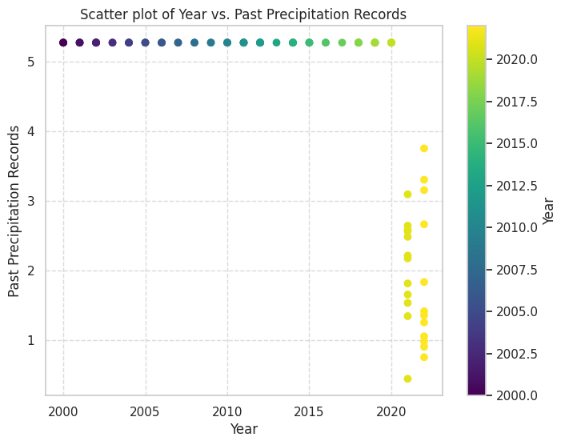
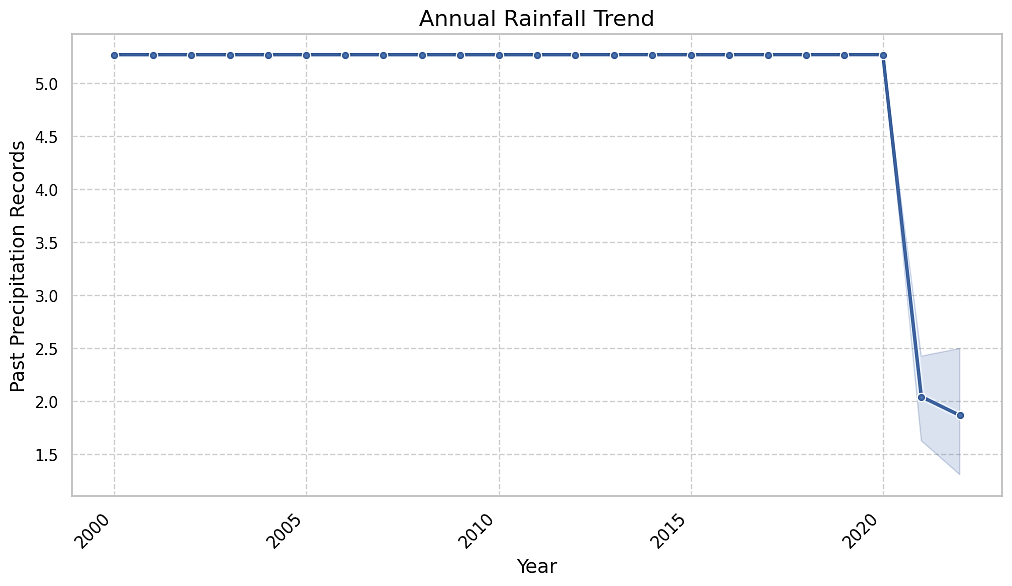
The outliers can lightly be seen in the humidity, where values are less than 78. To cater to this, the z-score method is utilized to remove outliers.

|  |
| --- |
| data\_filtered = data[data['RH2M Humidity'] >= 78]  z\_scores = stats.zscore(data.select\_dtypes(include=[float, int]))  abs\_z\_scores = abs(z\_scores)  filtered\_entries = (abs\_z\_scores < 3).all(axis=1)  data = data[filtered\_entries]  print(data.shape) |

*This reduces the dataset to 272 entries.*

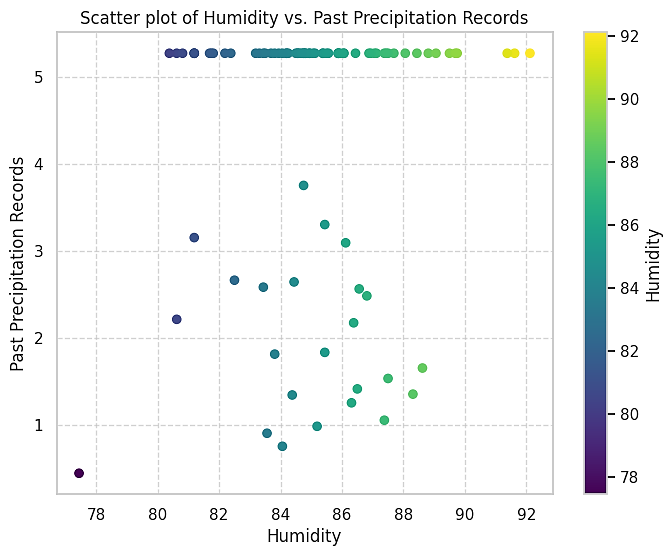
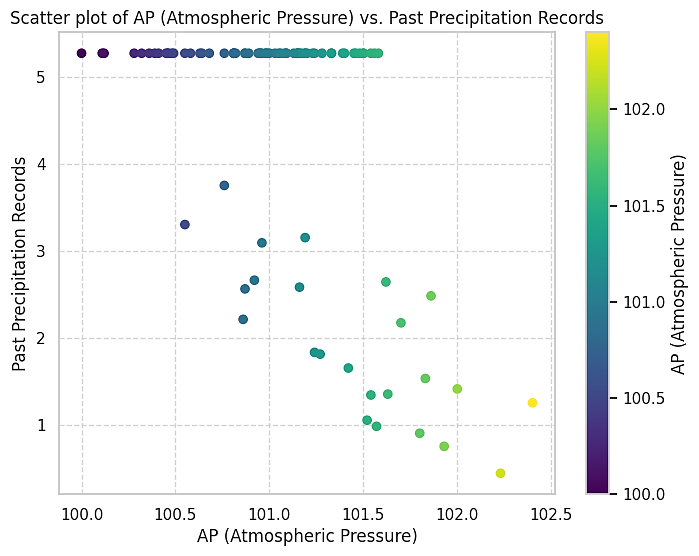
### Insightful Visualizations of the Existing Dataset

To understand the initial data better, the following visualizations are made



**Figure 3** The scatter plot for past precipitation records shows the same trend as above; the decrease in rainfall has been confirmed in recent years.

**Figure 2** Yearly past precipitation records indicate the rainfall annually.

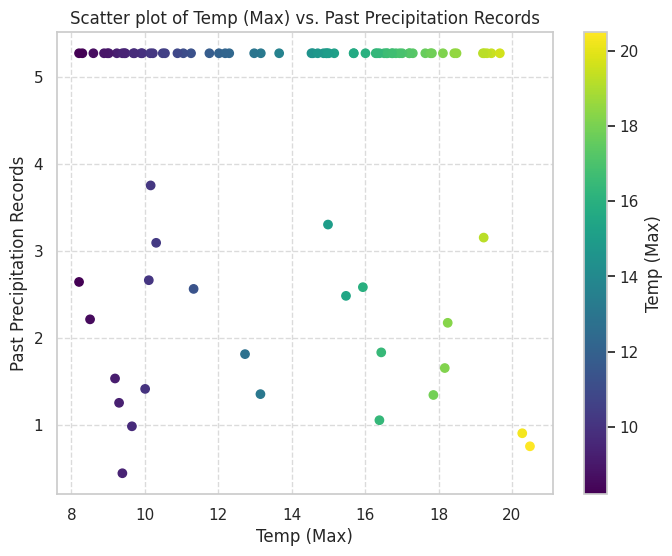
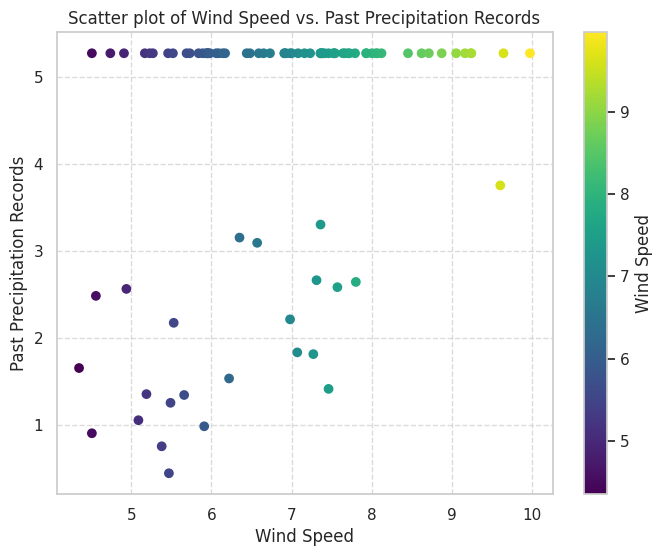


**Figure 4** **Yearly Change in Atmospheric Pressure.**

The fluctuations in atmospheric pressure over the years indicate seasonal changes. As Atmospheric pressure has an inverse relationship with rainfall, lower pressure often results in higher rainfall due to the formation of low-pressure systems, which can lead to precipitation.

**Figure 5: Yearly Change in Humidity**

The graph depicts annual variations in humidity levels. It shows periods of higher humidity, which may correspond with rainy seasons. High humidity levels indicate moisture-rich air, which can lead to precipitation.

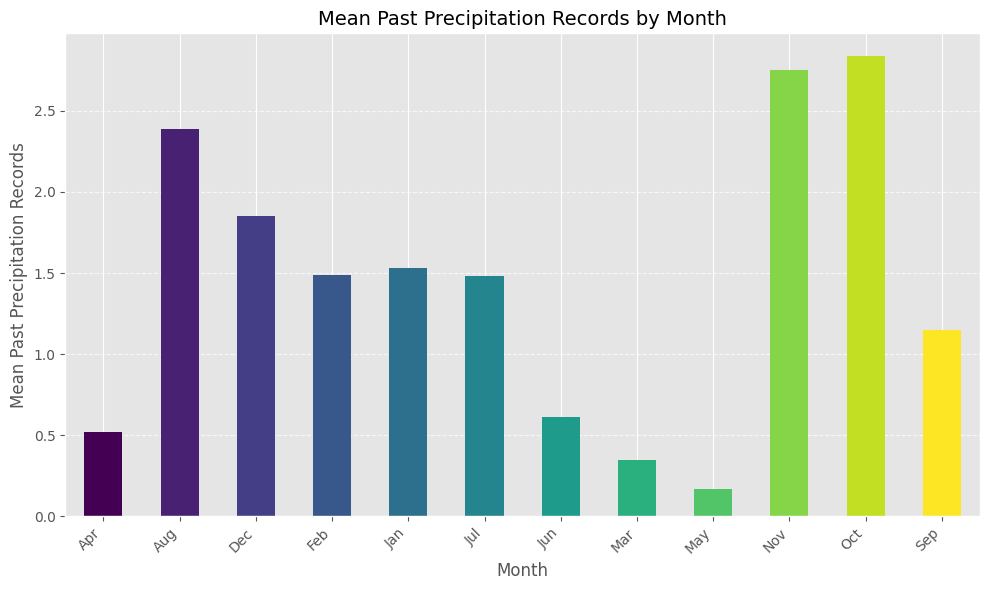
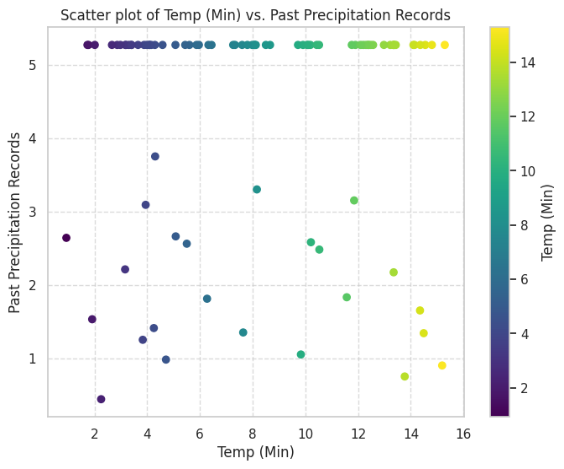


**Figure 6: Yearly Change in Wind Speed**

Wind speed varies yearly, with some years showing higher average wind speeds. Higher wind speeds can transport moisture-laden air masses, contributing to rainfall.

**Figure 7: Yearly Change in Maximum Temperature**

This graph shows fluctuations indicating warm and cool periods. Higher temperatures can increase evaporation rates, potentially leading to more rainfall if the moisture condenses.

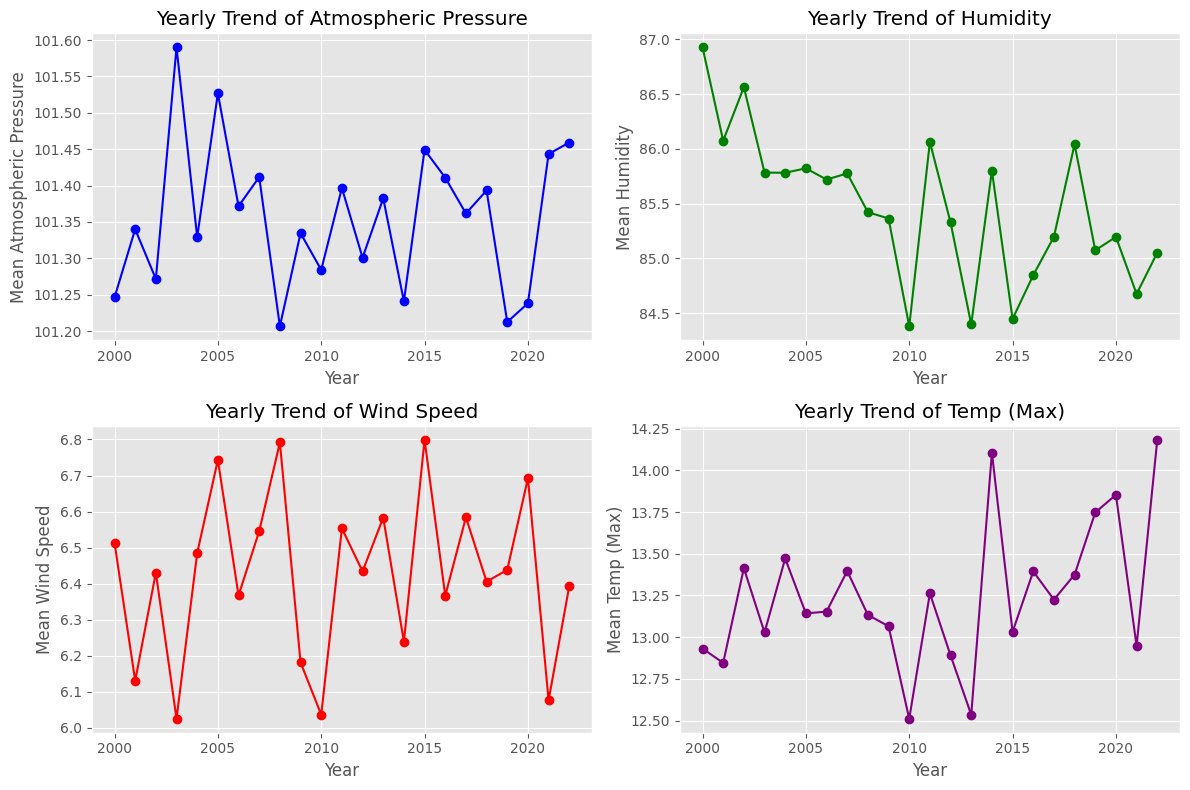


**Figure 9:**

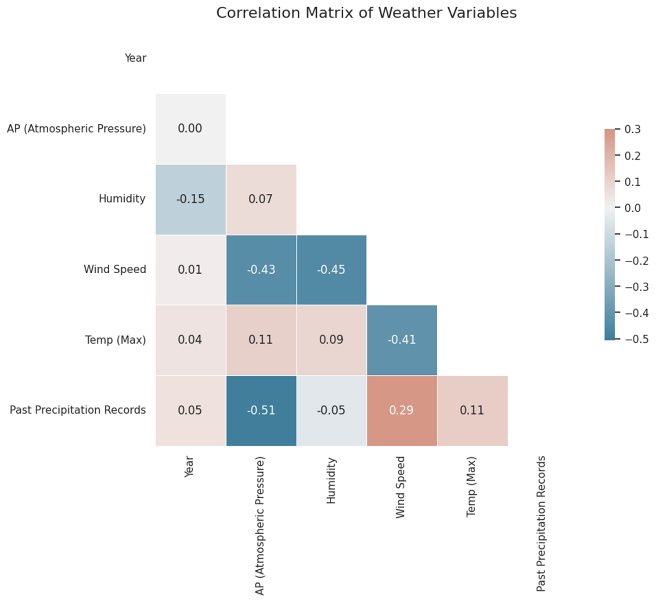
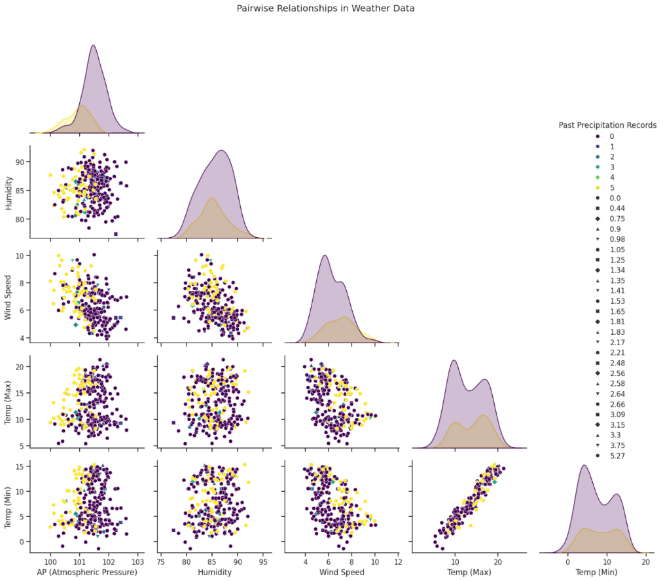
Rainfall is observed most in October and November throughout the year on an average

**Figure 8: Yearly Change in Minimum Temperature**

This graph shows periods of higher and lower minimum temperatures over the years. Higher minimum temperatures indicate a correlation with increased rainfall if they lead to greater dew point and moisture content in the air.



**Figure 10** Visualization of how the weather variables (each separately) change yearly.



**Figure 11:** **Relationships with Pair Plot between all features.**

Strong positive or negative correlations between temperature, humidity, and rainfall can guide feature selection for the prediction model.

**Figure 12:** **Correlation matrix of the features with a heatmap.**

The feature ‘Year’ shows a very weak correlation. Since Year is a significant feature to predict rainfall patterns yearly. It is decided not to drop ‘Year’

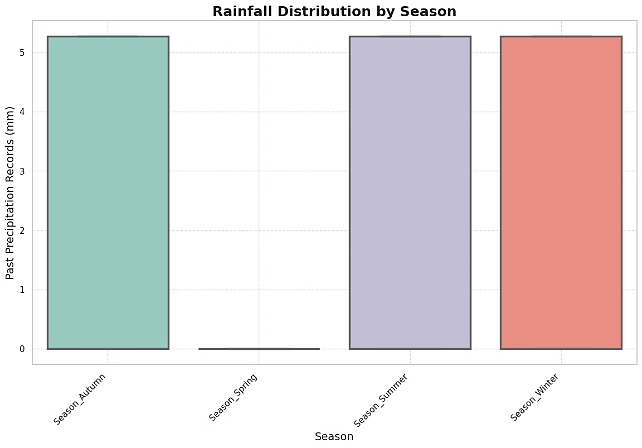
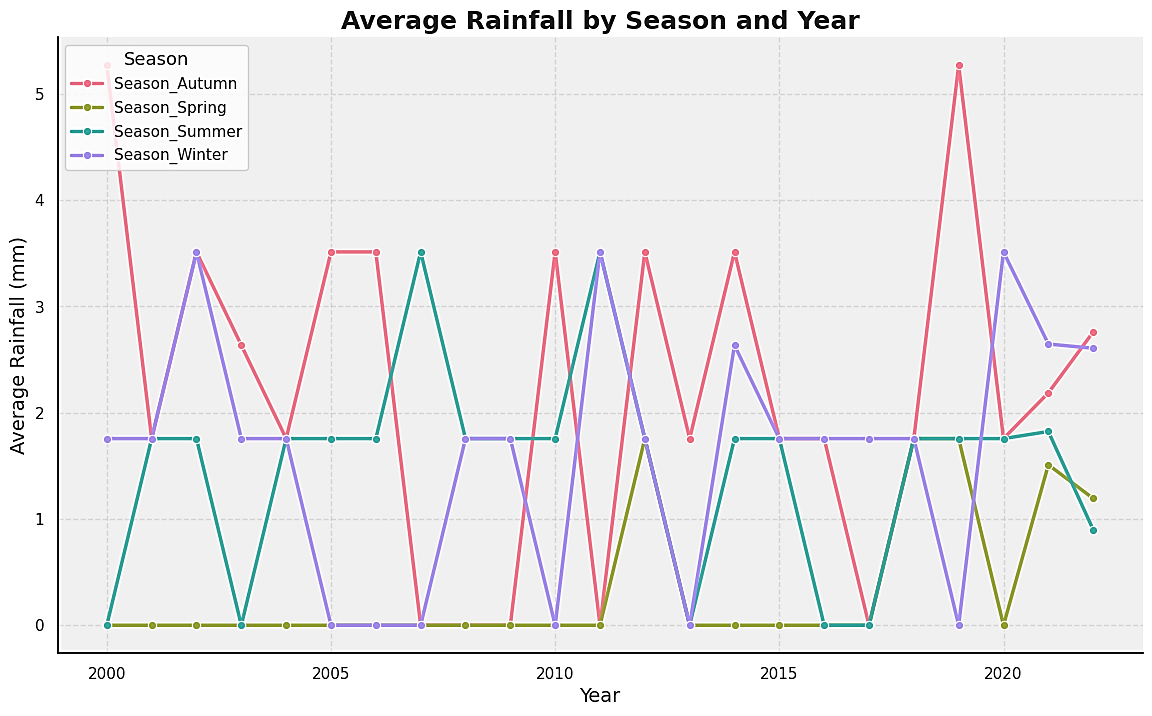
## Feature Engineering

Feature Engineering is carried out to improve the model’s predictive accuracy.

### 1 .Time-Based Features

For the time-based features, “Month” is converted to numeric values through one hot encoding, and variables of seasons are created. Such that:





**Figure 13:** The line plot indicates how average rainfall changes seasonally over the years. It shows which seasons tend to be wetter or drier. The autumn season consistently shows higher rainfall, whereas, the Spring season tends to be drier.

**Figure 14:** The box plot illustrates the distribution of rainfall across different seasons, showing median, quartiles, and outliers. Spring season to have the least rainfall confirmed.

### 2. Temperature Interactions with rainfall

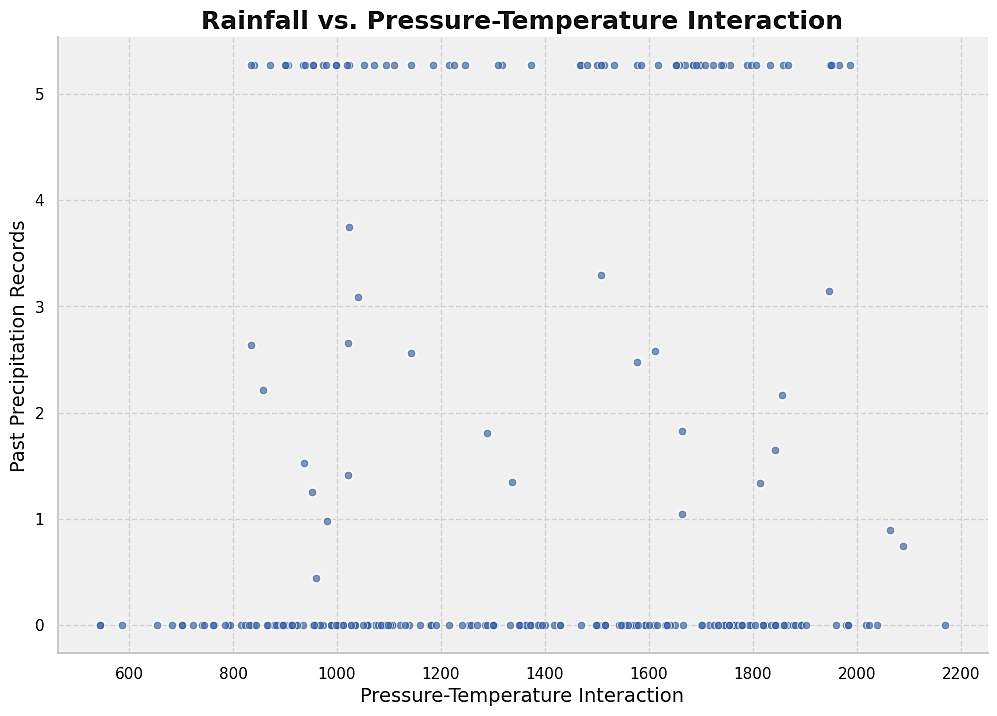
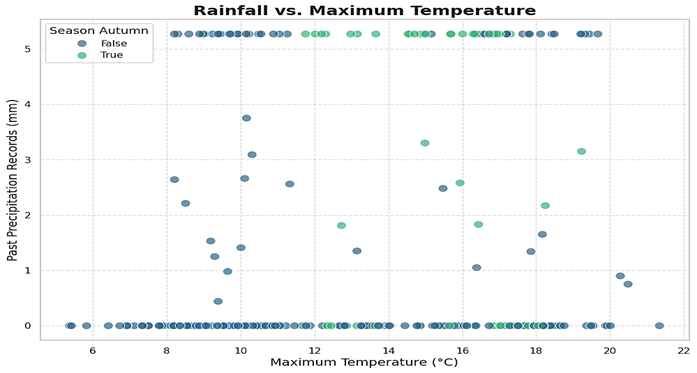
The temperature difference is calculated and stored in df['Temp\_Diff'] and then Pressure and Temperature Interaction is calculated

|  |
| --- |
| df['Temp\_Diff'] = df['Temp (Max)'] - df['Temp (Min)']  df['Pressure\_Temp\_Interaction'] = df['AP (Atmospheric Pressure)'] \* df['Temp (Max)'] |

Table 3 Demonstrating head of data frame

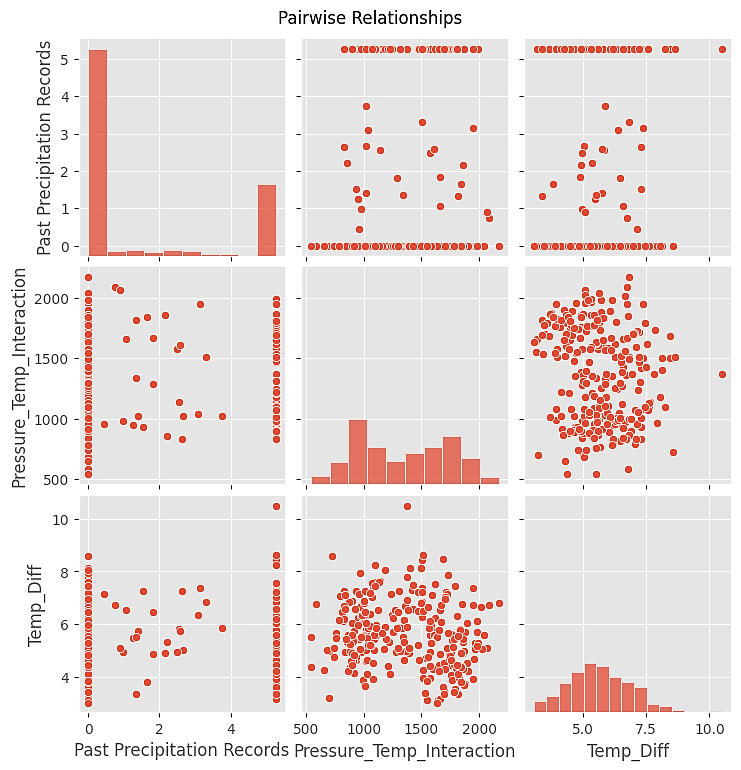
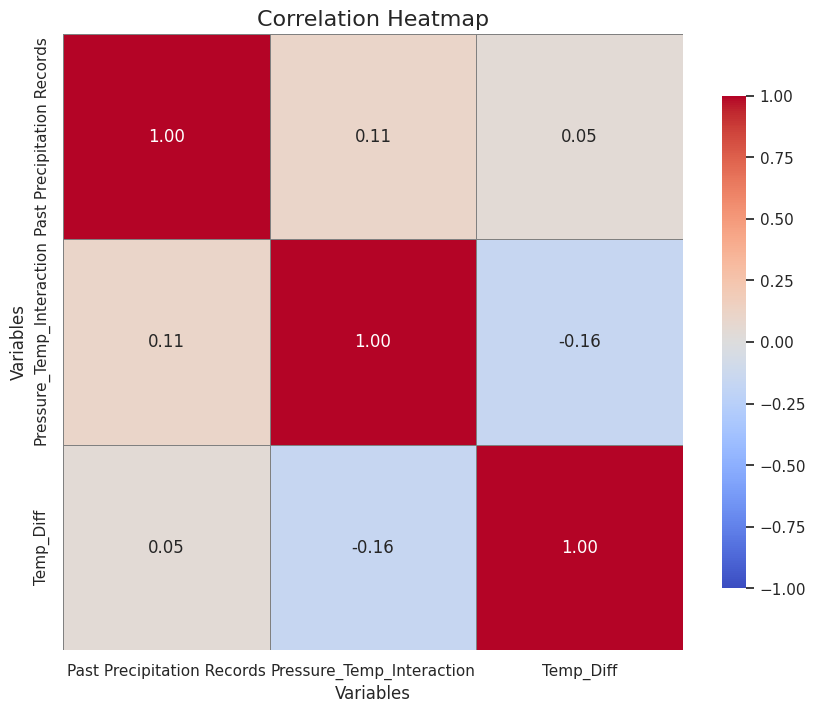


The following visualizations are made:



**Figure 15:** The box plot illustrates the distribution of rainfall in clusters indicating certain temperature ranges associated with higher rainfall.

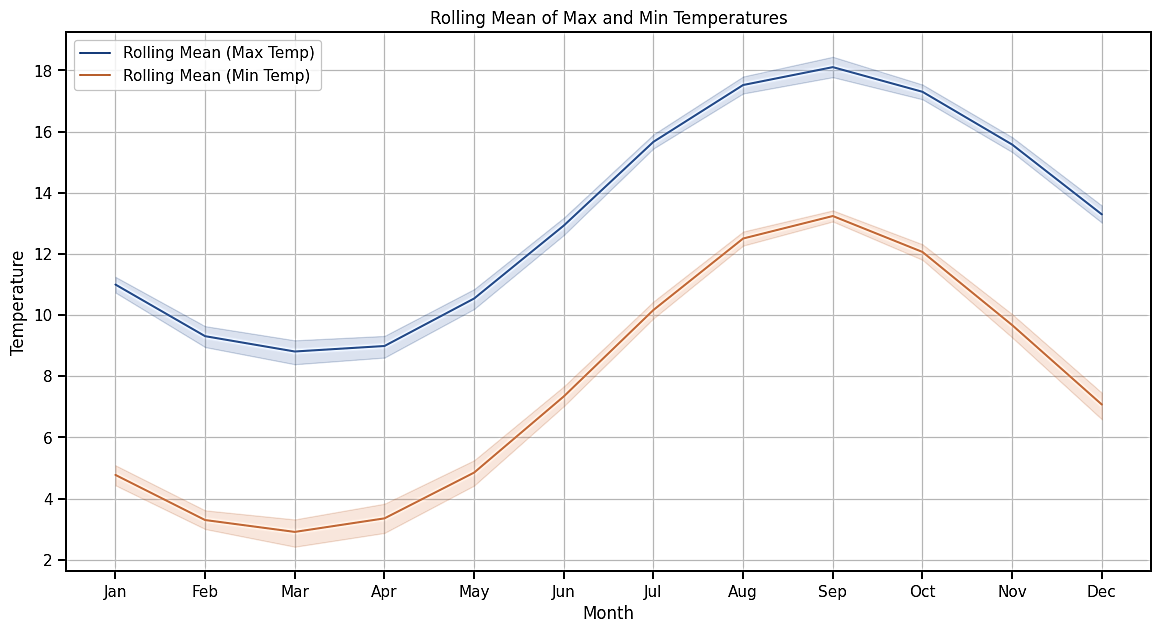
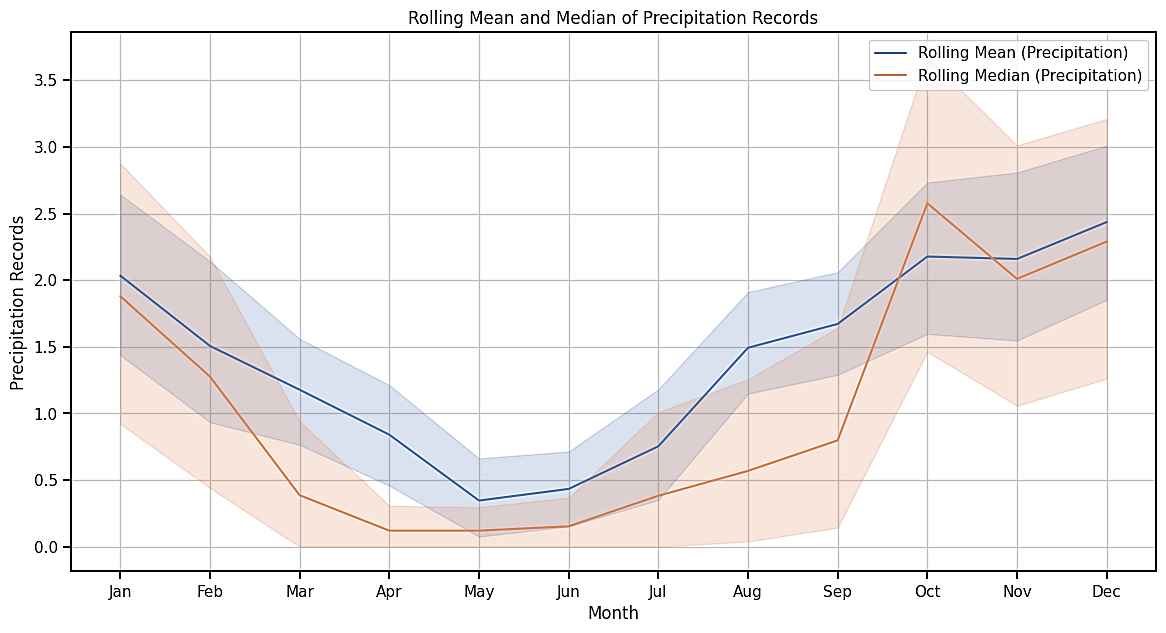
***Figure 16*** *Scatter plot of Rainfall and Temperature Interaction depicts clusters of points indicating different weather patterns such as monsoons, etc.*



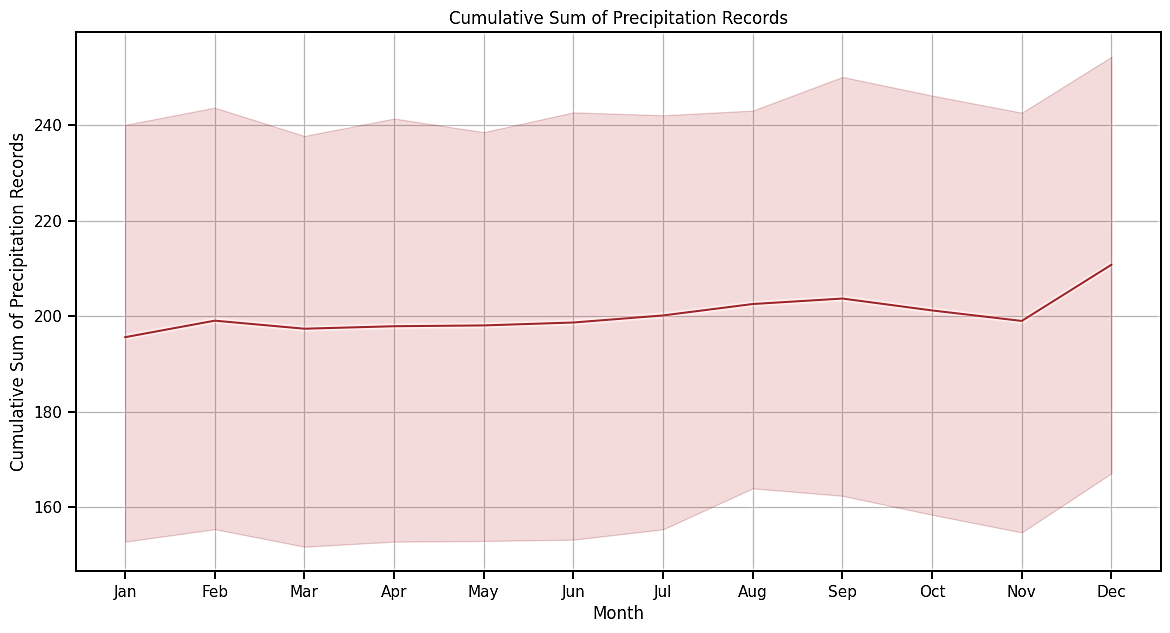
**Figure 17 & 18** Correlation Heatmap and Pairwise relationship between Pressure Temperature Interactions with the Rainfall, shows Temp (Diff) as poorly correlated

### 3. Rolling and Cumulative Features

The Rolling Mean smoothens the fluctuations and highlights longer-term trends for this feature engineering. Rolling Median reduces the impact of outliers, and lastly, a cumulative sum is calculated for past precipitation records.

 ***Figure 20 Rolling Mean of Temperature.*** *Both the maximum (blue line) and minimum (orange line) temperatures show a clear seasonal pattern. The gap between the maximum and minimum temperatures is wider during the warmer months (around month 8) and narrower during the cooler months (around month 1 and month 12).*

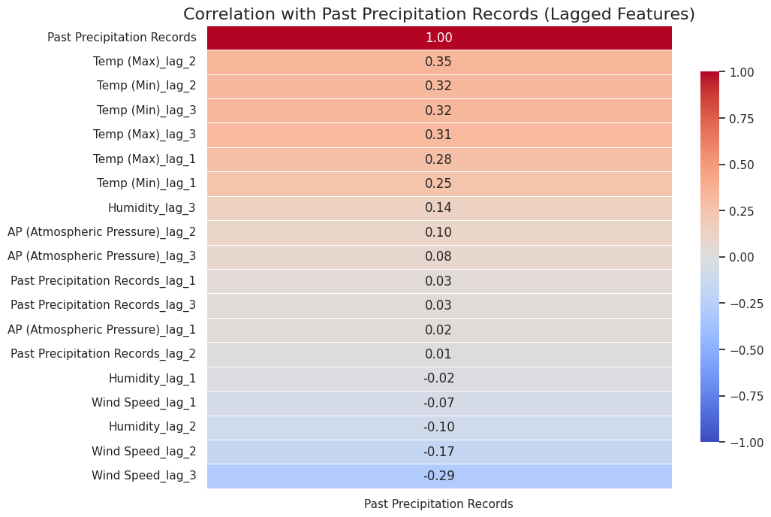
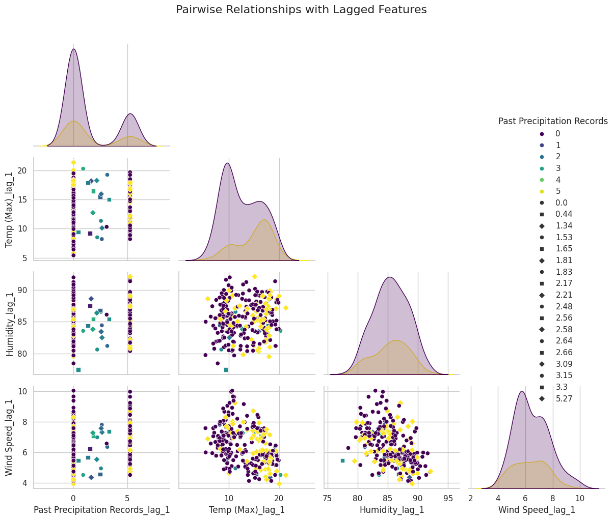
**Figure 19:** The rolling median (orange line) is typically lower than the rolling mean (blue line), indicating the presence of outliers (higher precipitation values) that affect the mean more than the median. Seasonal Trends: Both the mean and median show a decline in precipitation from month 1 to month 4 i.e. from Jan to April, followed by an increase towards month 12.



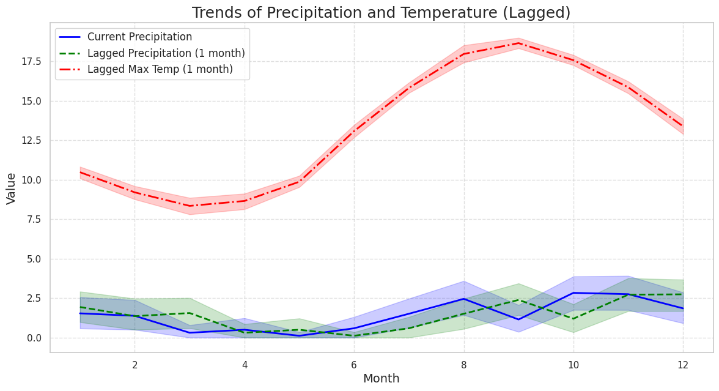
***Figure 21:*** *The cumulative sum starts around 200, fluctuates slightly, and increases towards the end of the year. This indicates periods of accumulation and stagnation in precipitation. The slight dip around months 4 to 6 suggests a period of lower precipitation, while the rise towards the end of the year suggests an increase. The shaded area shows the range of cumulative precipitation. The wider the shaded area, the more variability there is in monthly precipitation.*

### 4. Creating Lag Features (Lag Effects)

Lag features are created to capture temporal dependencies by shifting the values of the weather variables by a certain number of periods (e.g., the previous month's temperature, humidity, etc.). Its visualizations will help understand how they relate to the Past Precipitation Records. The following are the visualizations created:



***Figure 22 & 23*** *suggests* *that the temperature is a significant factor influencing precipitation records, with a positive relationship observed for lagged values of 1 to 3 days. The Wind Speed shows a negative relationship, particularly with a 3-day lag and Humidity and Atmospheric Pressure have moderate positive correlations, suggesting their potential impact on precipitation.*



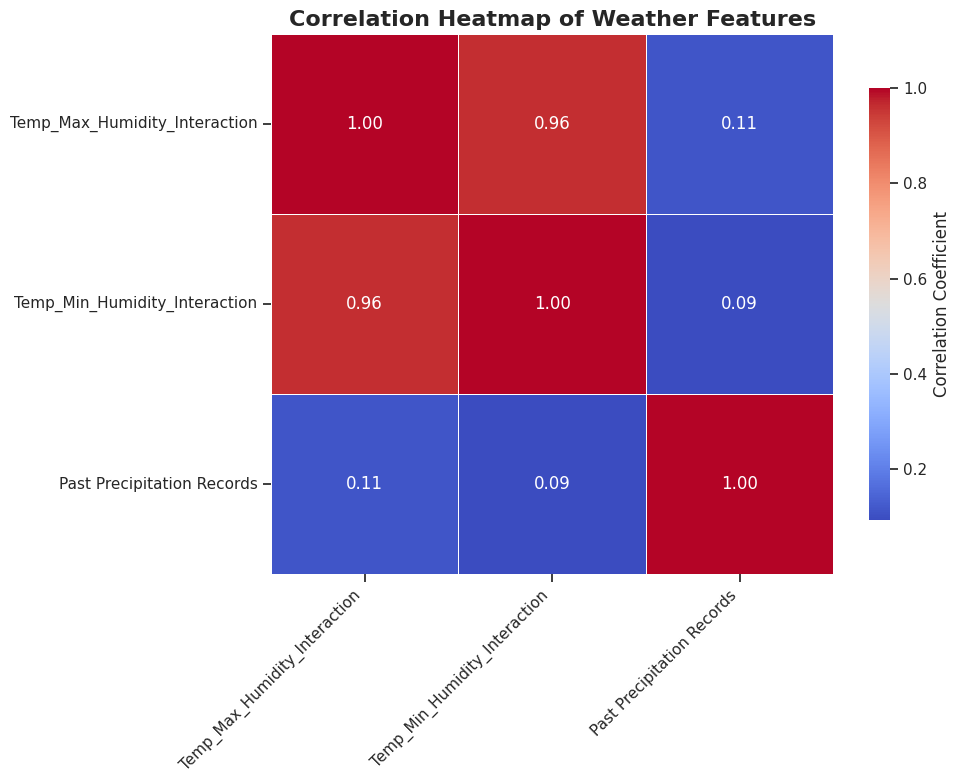
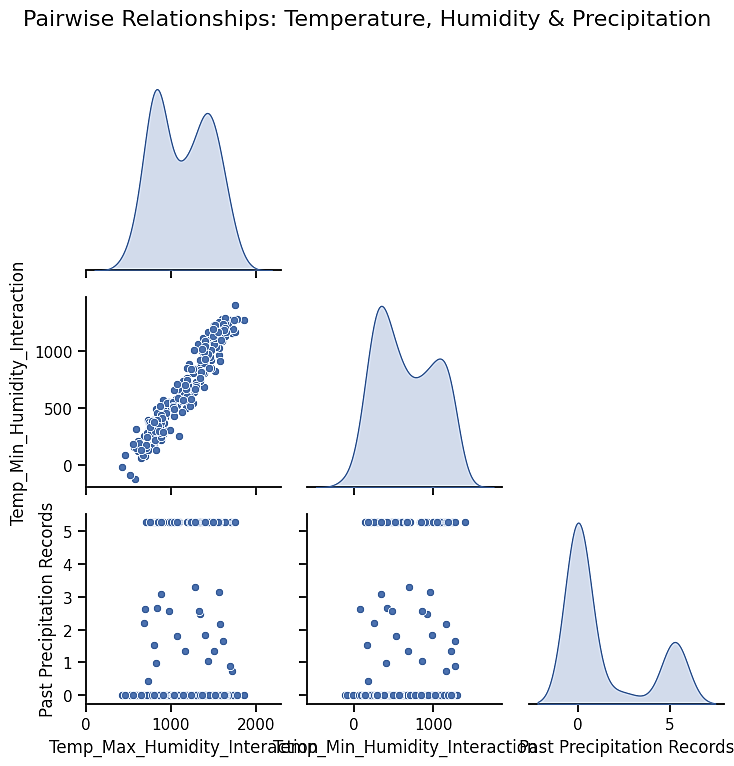
***Figure 24:*** *This graph illustrates the trends of current precipitation, lagged precipitation (1 month), and lagged maximum temperature (1 month) over a 12-month period.*

*The red dashed line (lagged maximum temperature), which shows a clear seasonal trend, peaking in the middle of the year (around July) and dipping towards the end of the year (December). The blue solid line (current precipitation levels) appears to fluctuate indicating variability in monthly rainfall. The green dashed line (lagged precipitation 1 month), closely follows the trend of the current precipitation but with a slight delay.*

### 5. Rain Analysis with Humidity and Temperature

To analyse the rain interactions with air temperature and humidity maximum and minimum temperature and humididty interactions have created

|  |
| --- |
| df['Temp\_Max\_Humidity\_Interaction'] = df['Temp (Max)'] \* df['Humidity']  df['Temp\_Min\_Humidity\_Interaction'] = df['Temp (Min)'] \* df['Humidity'] |

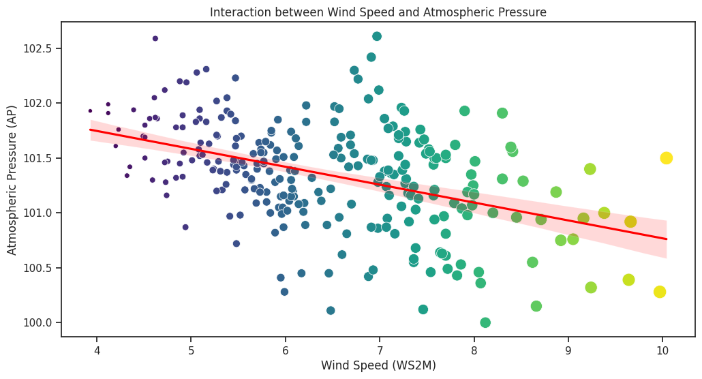


***Figure 25 & 26:*** *The scatter plot and (KDE) plots suggest a strong positive correlation between temperature (Max) and humidity interaction, also temperature (min) and humidity interaction. This indicates direct relationship with each other. There is a weak positive correlation between maximum temperature and humidity interaction and past precipitation records. This also indicates some instances of higher precipitation with varying temperature and humidity interaction. However, a weak correlation is detected in the heatmap with Temp\_Max\_Humidity\_Interaction (0.11) and Temp\_Min\_Humidity\_Interaction (0.09).*

### 6. Analysing Wind Speed and Atmospheric Effects on Rain

To analyse the rain interactions with Wind Speed and Atmospheric Pressure, following variables and visualizations have been generated:

|  |
| --- |
| df['WS\_AP\_Interaction'] = df['WS2M Wind Speed'] \* df['AP (Atmospheric Pressure)'] |

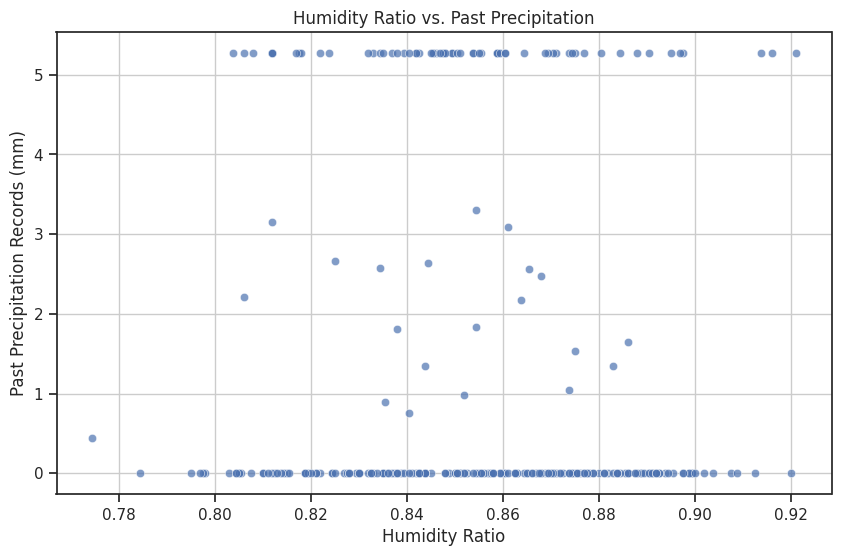
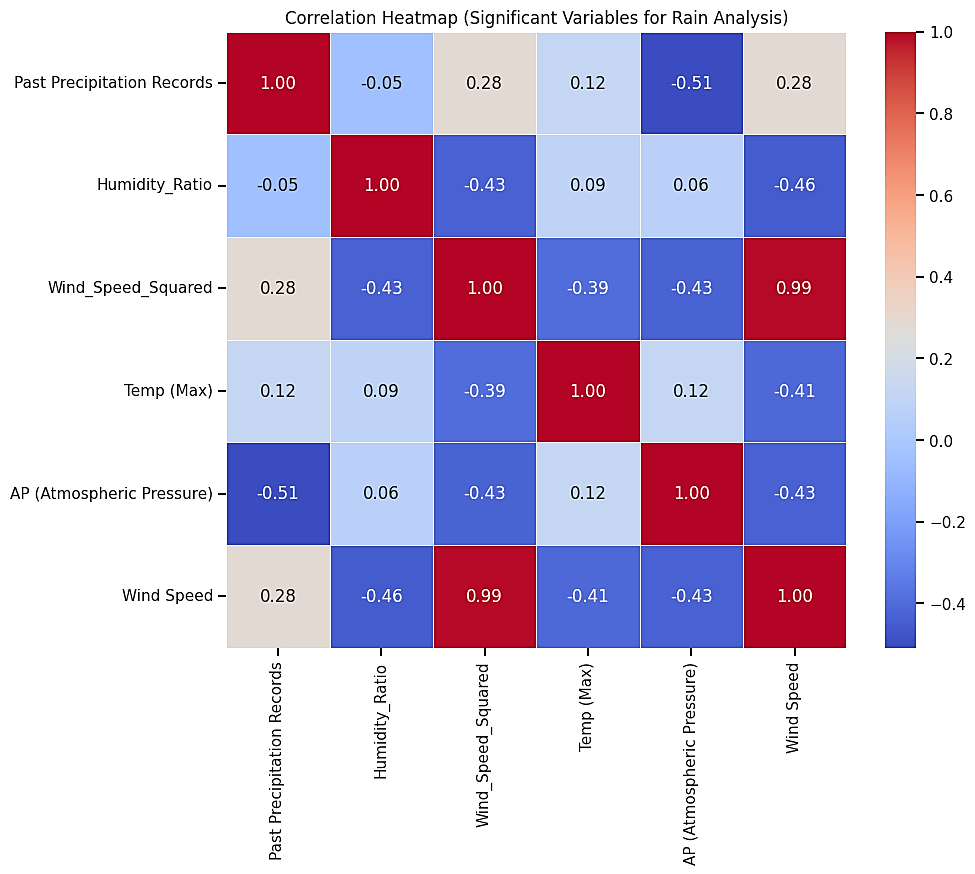


***Figure 27:*** *is a scatter plot with a regression line indicates a negative correlation between wind speed and atmospheric pressure. As wind speed increases, atmospheric pressure tends to decrease.*

### 7. Climate Indices

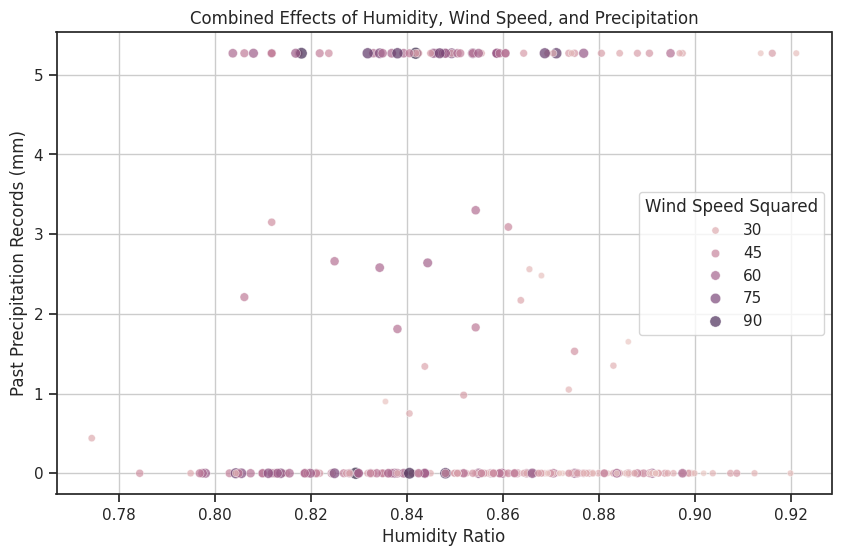
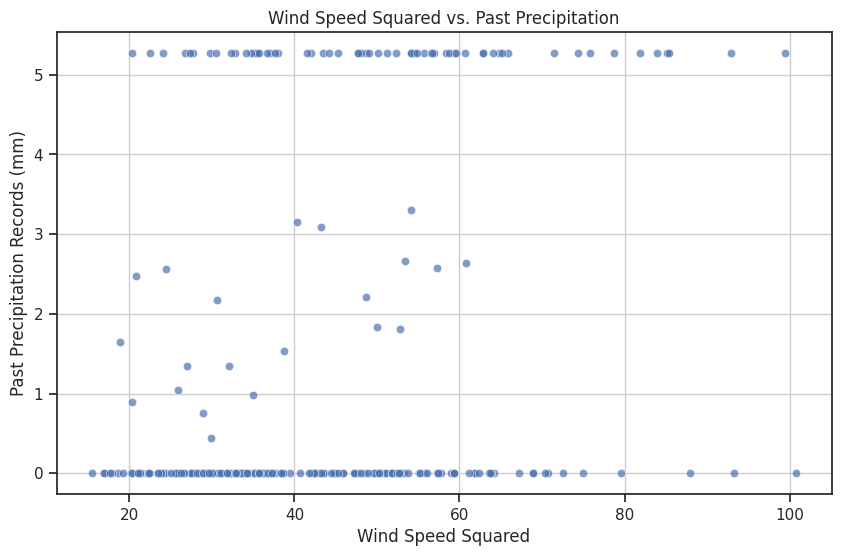
Climate indices are created to capture broader climatic conditions, such that the new variables formed are Temperature difference, Humidity ratio, and the square of Wind speed.

|  |
| --- |
| df['Temperature\_Range'] = df['Temp (Max)'] - df['Temp (Min)']  *# Assuming 100 is the maximum possible humidity*  df['Humidity\_Ratio'] = df['Humidity'] / 100  df['Wind\_Speed\_Squared'] = df['Wind Speed'] \*\* 2 |



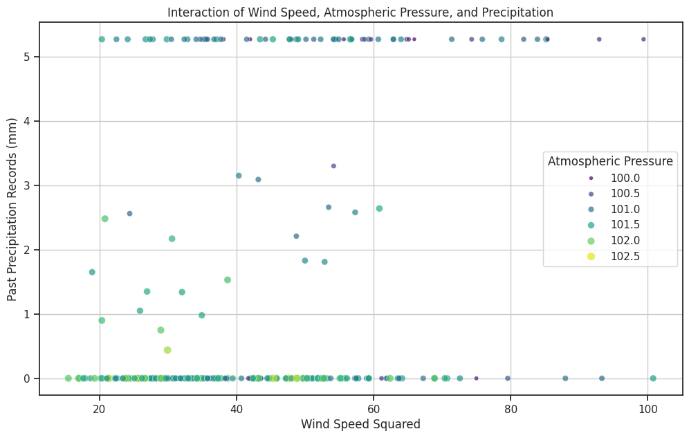
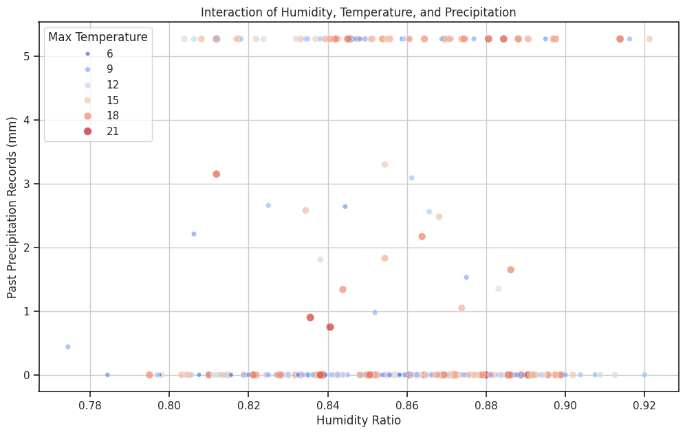
***Figure 28:*** *Past Precipitation Records has a positive correlation with Humidity\_Ratio (0.05), Wind\_Speed\_Squared (0.28), and Temp (Max) (0.12), and a negative correlation with AP (Atmospheric Pressure) (-0.51) and Wind Speed (0.28).*

***Figure 29:*** *The scatter plot indicates a very weak relationship between humidity ratio and past precipitation records. The majority of precipitation records are clustered at lower humidity ratios.*



***Figure 31:*** *The scatter plot shows the combined effect of humidity ratio and wind speed squared on past precipitation records. The majority of the data points are clustered at lower humidity ratios and wind speed squared values with some higher precipitation outliers.*

***Figure 30:*** *there is a weak relationship with most precipitation records being at lower wind speed squared values. There are a few outliers indicating higher precipitation at higher wind speed squared values.*



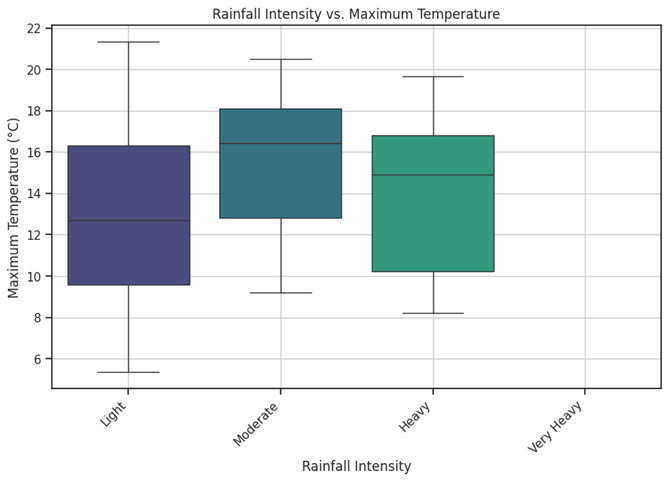
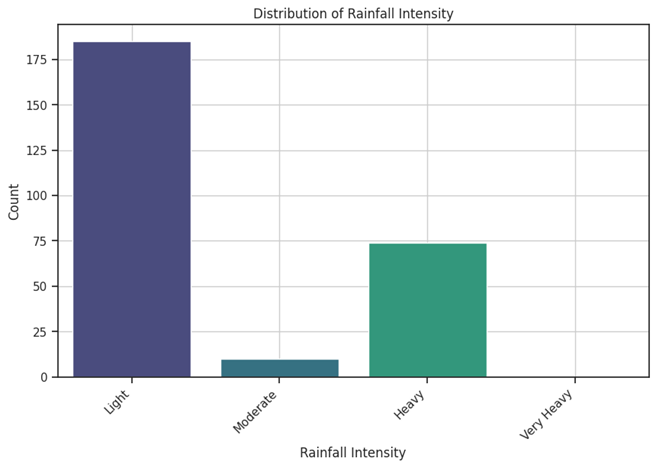
***Figure 32:*** *This plot highlights the interaction between humidity ratio, maximum temperature, and past precipitation records. The majority of precipitation records are concentrated at lower humidity ratios and various temperature levels, indicating a weak overall relationship with some notable outliers.*

***Figure 33:*** *The scatter plot depicts the interaction between wind speed squared, atmospheric pressure, and past precipitation records. The data indicates that most precipitation events occur at lower wind speed squared values and various atmospheric pressure levels, with some higher precipitation events occurring at higher wind speed squared values and varying atmospheric pressures.*

### 8. Rainfall Intensity (amount of rain per hour)

New featuresets for rain intensity categories are made such as Light, Moderate, Heavy, and Very Heavy

|  |
| --- |
| bins = [0, 0.5, 2, 10, np.inf] # Define intensity categories  labels = ['Light', 'Moderate', 'Heavy', 'Very Heavy']  df['Rainfall\_Intensity'] = pd.cut(df['Past Precipitation Records'], bins=bins, labels=labels, right=False) |

****

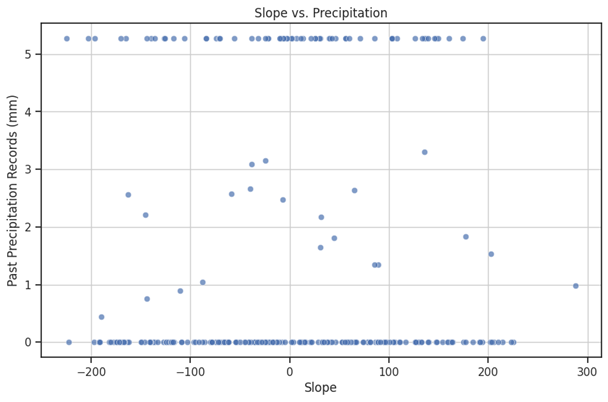
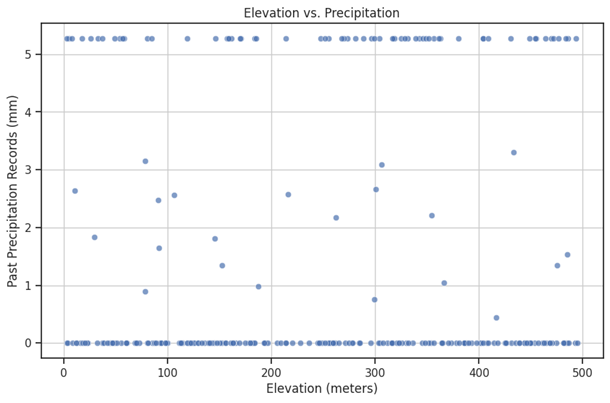
***Figure 35:*** *Light rainfall is associated with the widest range of maximum temperatures. Moderate and heavy rainfall tend to occur at lower maximum temperatures, with heavy rainfall and very heavy rainfall occurs at the lowest maximum temperatures among the four categories, suggesting that extreme rainfall is linked to cooler conditions.*

***Figure 34****: The majority of rainfall events in Cumbria are categorized as Light. Moderate rainfall events are fewer, followed by "Heavy" and "Very Heavy" rainfall events, which are rare.*

### 9. Topographical Features (Data Elevation)

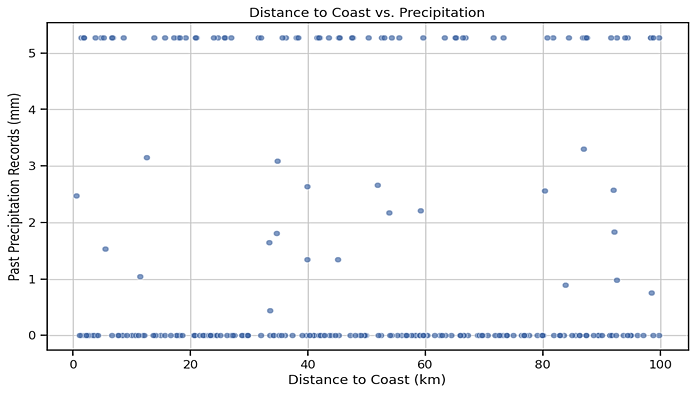
Simulation of the data to elevation data is done. New features like Elevation, Latitude, Longitude, Average and Range of Elevation, and slope is calculated.

|  |
| --- |
| np.random.seed(42)  df['Elevation'] = np.random.uniform(0, 500, len(df)) # Replace with your elevation data  df['Latitude'] = np.random.uniform(30, 40, len(df)) # Replace with your latitude data  df['Longitude'] = np.random.uniform(-120, -110, len(df)) # Replace with your longitude data  df['Average\_Elevation'] = df['Elevation'].mean()  df['Elevation\_Range'] = df['Elevation'].max() - df['Elevation'].min()  # This assumes you have elevation data for neighboring points to calculate slope  df['Slope'] = np.gradient(df['Elevation'])  df['Distance\_to\_Coast'] = np.random.uniform(0, 100, len(df)) |

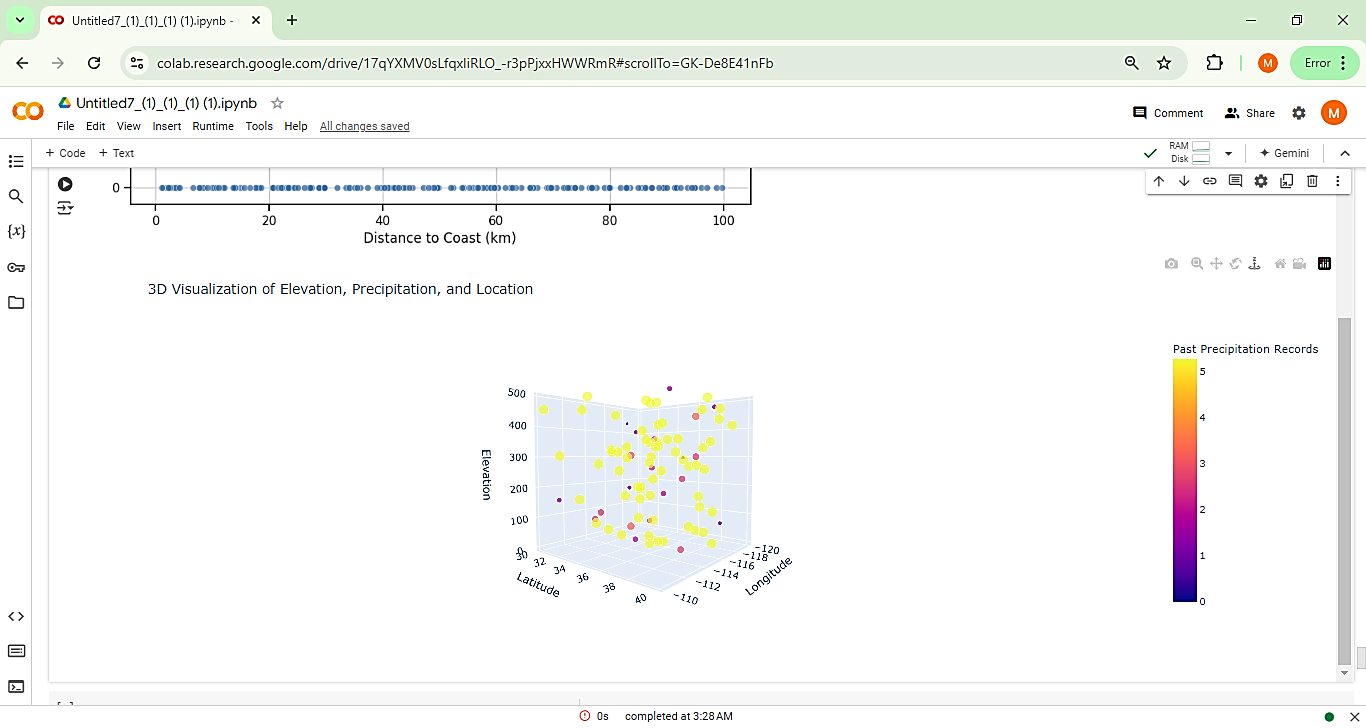


***Figure 36:*** *There is no clear trend that indicate an increase or decrease in precipitation with elevation. Most of the instances of higher precipitation across all elevation ranges, suggests that rainfall in Cumbria is not strongly influenced by elevation. However, some higher precipitation events at higher elevations, indicates localized weather patterns or orographic effects (where moist air is lifted over mountains, leading to precipitation).*

***Figure 37:*** *As the majority of the data points cluster around a slope of 0, a scattering of precipitation records across a wide range of slopes are observed with no clear pattern or trend indicating a strong relationship between slope and precipitation. This suggests that slope may not be a significant factor influencing rainfall distribution in Cumbria.*



***Figure 37:*** *No clear pattern or trend is visible in the scatter plot, indicating that precipitation events are distributed fairly evenly regardless of the distance to the coast. This suggests that coastal proximity does not have a strong influence on rainfall intensity in Cumbria. Other factors, such as elevation, weather systems, and local topography, might play a more significant role in determining rainfall patterns.*



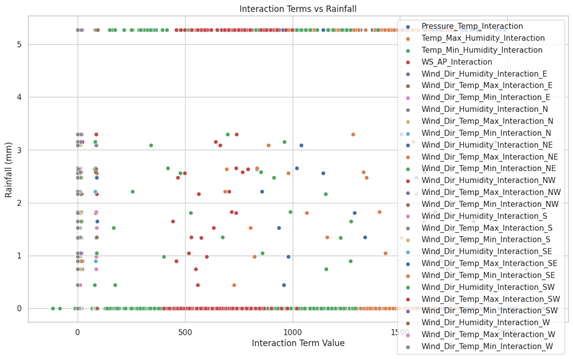
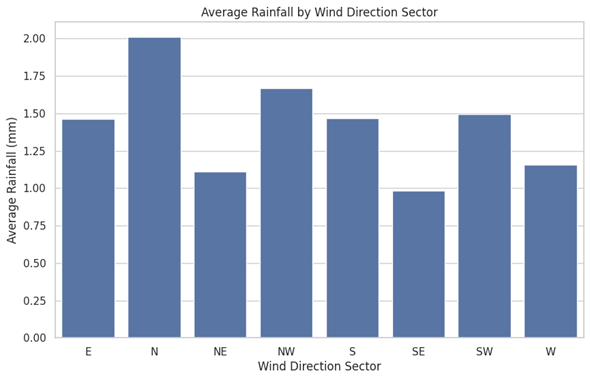
**Figure 38** Shows a 3D representation of Elevation and Precipitation by Location (Longitude, Latitude)

### 10. Wind Direction

To analyze the relationship with rainfall of the wind direction and create interaction terms, the code is first the code performs several tasks. Such that,

|  |
| --- |
| # A synthetic 'Wind Direction' data is created with the values between 0 and 360 degrees for each # row in the DataFrame `df`.  np.random.seed(0)  df['Wind Direction'] = np.random.uniform(0, 360, len(df))  ```  # Wind direction categories are defined  def categorize\_wind\_direction(degrees):  if degrees >= 337.5 or degrees < 22.5:  return 'N'  elif degrees < 67.5:  return 'NE'  elif degrees < 112.5:  return 'E'  elif degrees < 157.5:  return 'SE'  elif degrees < 202.5:  return 'S'  elif degrees < 247.5:  return 'SW'  elif degrees < 292.5:  return 'W'  else:  return 'NW'  df['Wind Direction Sector'] = df['Wind Direction'].apply(categorize\_wind\_direction)  # This function categorizes the wind direction into sectors (e.g., N, NE, E) based on the degree values.  # The data is grouped by wind direction sector and calculates the mean rainfall for each sector. rainfall\_by\_wind\_dir = df.groupby('Wind Direction Sector')['Past\_Precipitation\_Records'].mean().reset\_index()  print(rainfall\_by\_wind\_dir)  # Create interaction terms between wind direction categories and other features  wind\_dir\_dummies = pd.get\_dummies(df['Wind Direction Sector'])  for col in wind\_dir\_dummies.columns:  df[f'Wind\_Dir\_Humidity\_Interaction\_{col}'] = df['RH2M\_Humidity'] \* wind\_dir\_dummies[col]  df[f'Wind\_Dir\_Temp\_Max\_Interaction\_{col}'] = df['Temp\_Max'] \* wind\_dir\_dummies[col]  df[f'Wind\_Dir\_Temp\_Min\_Interaction\_{col}'] = df['Temp\_Min'] \* wind\_dir\_dummies[col] |

The following visualizations are created:



***Figure 39:*** *This scatter plot suggest that certain combinations of wind direction and temperature have a significant impact on rainfall in Cumbria.*

***Figure 38:*** *The bar chart shows average rainfall in different wind direction sectors. The N (North) has the highest average rainfall. NE (Northeast) has the lowest average rainfall. Other sectors (E, NW, S, SE, SW, W) have moderate rainfall, with NW and SW having relatively higher averages.*

# Important Feature Selecting, Dropping Irrelevant Features

The code selects and drops features using two main methods:

1. ***Correlation Analysis***

This calculates the correlation of numeric features with the target variable and here we’ve dropped features with correlation values below a threshold (0.1).

1. ***Random Forest Feature Importance***

This train a Random Forest model to determine feature importance and drop features with importance values below a threshold (0.01).

Additionally, specific features are manually selected for further analysis:

|  |
| --- |
| features = ['Temp\_Max', 'RH2M\_Humidity', 'AP\_Atmospheric\_Pressure', 'WS2M\_Wind\_Speed', 'Month']  X = data[features]  y = data['Past\_Precipitation\_Records'] |

# 2. Model Development

## Algorithm Selection

The following models chosen are suitable for predicting rainfall, allowing comparison of their performances:

***Linear Regression:*** Trained a linear regression model.

***Ridge Regression:*** Trained a ridge regression model.

***Random Forest Regressor:*** Trained a random forest regression model.

## Dataset Splitting

The dataset is divided into training and testing subsets using an 80-20 split to evaluate model performance effectively:

|  |
| --- |
| X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42) |

## Model Training

### Linear Regression

|  |
| --- |
| lr\_model = LinearRegression()  lr\_model.fit(X\_train, y\_train)  lr\_predictions = lr\_model.predict(X\_test) |

### Ridge Regression

|  |
| --- |
| ridge\_model = Ridge()  ridge\_model.fit(X\_train, y\_train)  ridge\_predictions = ridge\_model.predict(X\_test) |

### Random Forest Regressor

|  |
| --- |
| rf\_model = RandomForestRegressor()  rf\_model.fit(X\_train, y\_train)  rf\_predictions = rf\_model.predict(X\_test) |

## Feature Importance

Random Forest model identified feature importance:

|  |
| --- |
| importance\_df = pd.DataFrame({'Feature': X.columns, 'Importance': feature\_importance})  importance\_df = importance\_df.sort\_values(by='Importance', ascending=False) |

**Table 4 Feature importance**

|  |  |
| --- | --- |
| Feature | Importance |
| AP\_Atmospheric\_Pressure | 0.411436 |
| Temp\_Max | 0.197236 |
| WS2M\_Wind\_Speed | 0.159693 |
| RH2M\_Humidity | 0.141927 |
| Month | 0.089707 |

*No features were deemed to have low importance (below the threshold of 0.01):*

|  |
| --- |
| print("Low importance features:\n", low\_importance\_features.tolist()) |

# Model Evaluation

## Performance Metrics

### Regression Tasks

Models are evaluated using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE):

|  |
| --- |
| def evaluate\_model(predictions, y\_test):  mae = mean\_absolute\_error(y\_test, predictions)  rmse = np.sqrt(mean\_squared\_error(y\_test, predictions))  return mae, rmse |

### Evaluations

|  |
| --- |
| lr\_mae, lr\_rmse = evaluate\_model(lr\_predictions, y\_test)  ridge\_mae, ridge\_rmse = evaluate\_model(ridge\_predictions, y\_test)  rf\_mae, rf\_rmse = evaluate\_model(rf\_predictions, y\_test) |

### Results

*Linear Regression* - MAE: 1.4922, RMSE: 1.7667

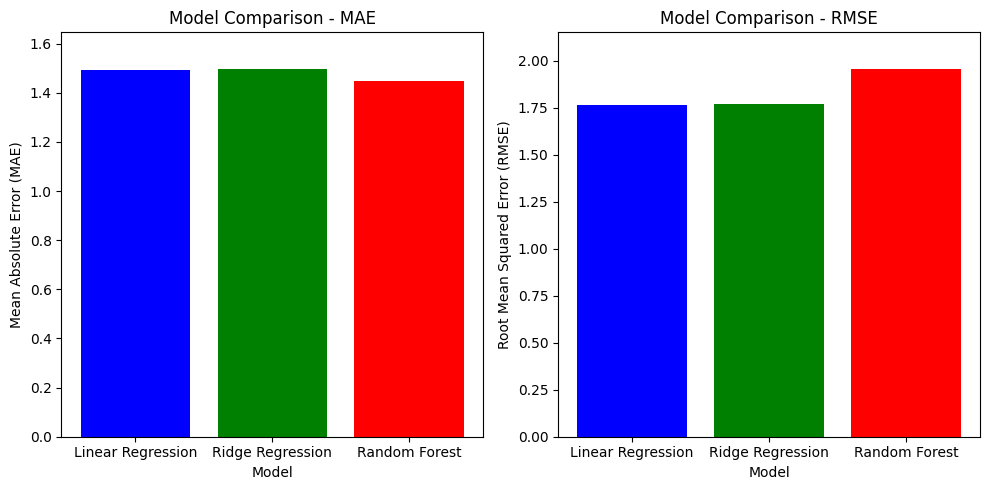
*Ridge Regression* - MAE: 1.4962, RMSE: 1.7692

*Random Forest* - MAE: 1.4492, RMSE: 1.9555

### Selecting the Best Model

|  |
| --- |
| best\_model = None  best\_rmse = min(lr\_rmse, ridge\_rmse, rf\_rmse)    if best\_rmse == lr\_rmse:  best\_model = 'Linear Regression'  elif best\_rmse == ridge\_rmse:  best\_model = 'Ridge Regression'  else:  best\_model = 'Random Forest' |

*Best Model Saved as Linear Regression*



***Figure 40*** ***Model Comparison - MAE & RMSE***

*All three models in MAE have a similar Mean Absolute Error (MAE) of around 1.4 to 1.5. This indicates that on average, the prediction error of rainfall using these models is between 1.4 and 1.5 mm and the RMSE for all models is around 1.75 to 2.0. RMSE penalizes larger errors more than MAE, and the values suggest that there are some larger errors in predictions, but overall performance is similar across these models.*

### Classification Tasks

Binary classification is performed with Logistic Regression and Random Forest Classifier:

|  |
| --- |
| log\_reg\_model = LogisticRegression()  log\_reg\_model.fit(X\_train, y\_train\_binary)  log\_reg\_predictions = log\_reg\_model.predict(X\_test)    rf\_clf\_model = RandomForestClassifier()  rf\_clf\_model.fit(X\_train, y\_train\_binary)  rf\_clf\_predictions = rf\_clf\_model.predict(X\_test) |

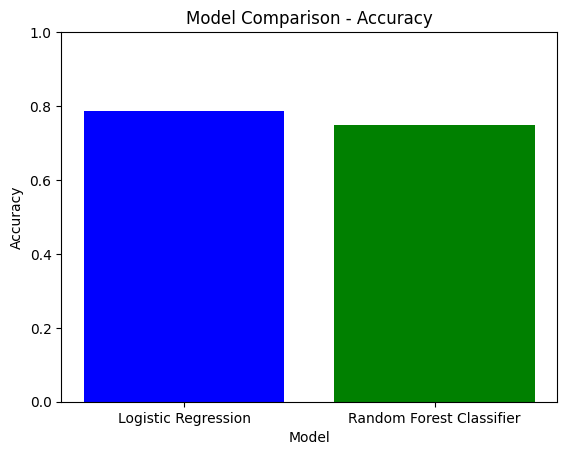
### Evaluation Metrics

|  |
| --- |
| def evaluate\_classification\_model(predictions, y\_test):  accuracy = accuracy\_score(y\_test, predictions)  precision = precision\_score(y\_test, predictions)  recall = recall\_score(y\_test, predictions)  return accuracy, precision, recall    log\_reg\_accuracy, log\_reg\_precision, log\_reg\_recall = evaluate\_classification\_model(log\_reg\_predictions, y\_test\_binary)  rf\_clf\_accuracy, rf\_clf\_precision, rf\_clf\_recall = evaluate\_classification\_model(rf\_clf\_predictions, y\_test\_binary)  print(f"Logistic Regression - Accuracy: {log\_reg\_accuracy:.4f}, Precision: {log\_reg\_precision:.4f}, Recall: {log\_reg\_recall:.4f}")  print(f"Random Forest Classifier - Accuracy: {rf\_clf\_accuracy:.4f}, Precision: {rf\_clf\_precision:.4f}, Recall: {rf\_clf\_recall:.4f}") |

### Results

*Logistic Regression* - Accuracy: 0.7857, Precision: 0.8000, Recall: 0.5714

*Random Forest Classifier* - Accuracy: 0.7500, Precision: 0.6842, Recall: 0.6190



***Figure 41*** ***Model Comparison - Accuracy***

*The Logistic Regression has slightly higher accuracy compared to Random Forest Classifier. This suggests that the Logistic Regression is slightly better at correctly predicting the categorical outcome related to rainfall.*

## Optimization

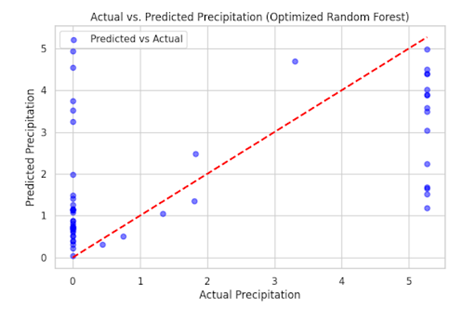
### Hyperparameter Tuning

Hyperparameter tuning is performed using GridSearchCV on the Random Forest model:

|  |
| --- |
| param\_grid = {  'n\_estimators': [50, 100, 200],  'max\_depth': [None, 10, 20, 30],  'min\_samples\_split': [2, 5, 10],  'min\_samples\_leaf': [1, 2, 4]  }    grid\_search = GridSearchCV(estimator=RandomForestRegressor(), param\_grid=param\_grid, cv=5, scoring='neg\_mean\_squared\_error', n\_jobs=-1, verbose=2)  grid\_search.fit(X\_train, y\_train)    best\_params = grid\_search.best\_params\_  best\_score = -grid\_search.best\_score\_    print(f'Best Parameters: {best\_params}')  print(f'Best Score (MSE): {best\_score}') |

### Evaluate Optimized Model

|  |
| --- |
| best\_rf\_model = grid\_search.best\_estimator\_  y\_pred\_best\_rf = best\_rf\_model.predict(X\_test)  mae\_best\_rf = mean\_absolute\_error(y\_test, y\_pred\_best\_rf)  rmse\_best\_rf = np.sqrt(mean\_squared\_error(y\_test, y\_pred\_best\_rf))    print(f'Optimized Random Forest MAE: {mae\_best\_rf}')  print(f'Optimized Random Forest RMSE: {rmse\_best\_rf}') |



***Figure 42*** ***Actual Vs Predicted Precipitation***

*The Actual vs. Predicted Precipitation plot for the optimized Random Forest model demonstrates that it has good predictive power, with most predictions being close to the actual observed values, although occasional large deviations indicate areas for further model refinement.*

# Conclusion

In summary, for predicting rainfall intensity categories, Logistic Regression was the most effective model, while for continuous rainfall prediction, the Random Forest Regressor demonstrated the highest accuracy.