A Micro Project report on

Develop a predictive model to study the rainfall of your society.

Submitted to the CMR Institute of Technology, Hyderabad in partial fulfilment of the requirement for the award of the Laboratory of

Artificial Intelligence and Machine Learning

of

III-B.Tech. I-Semester

in

Computer Science and Engineering

Submitted by

A. Priaanka	**23R01A05CR**
A. Siri	**23R01A05CT**
A. Niranjan	**23R01A05CU**

Under the Guidance Of

Mrs.A.Radhika

(Professor, Dept of CSE)

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMUS)

Approved by AICTE, Permanently Affiliated to JNTU, Hyderabad,

Accredited by NBA and NAAC with A+Grade.

Kandlakoya(V), Medchal Dist 501401

2025-2026

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMUS)
Approved by AICTE, Permanently Affiliated to JNTU, Hyderabad,
Accredited by NBA and NAAC with A+Grade.
Kandlakoya(V), Medchal Dist 501401

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that a Micro Project entitled with: "Develop a predictive model to study the rainfall of your society" is being

Submitted By

A. Priaanka	**23R01A05CR**
A. Siri	**23R01A05CT**
A. Niranjan	**23R01A05CU**

In partial fulfillment of the requirement for award of the Artificial Intelligence and Machine Learning of III-B.Tech I- Semester in CSE towards a record of a bonafide work carried out under our guidance and supervision.

Signature of Faculty

Mrs. A. Radhika
(Assistant Professor)

Signature of HOD

Dr. K. Pradeep Reddy
(Head of Department)

ACKNOWLEDGEMENT

We are extremely grateful to **Dr. M. Janga Reddy**, **Director**, **Dr. B.Satyanarayana**, **Principal** and **Dr. K. Pradeep Reddy**, **Head of Department**, Dept of Computer Science and Engineering, CMR Institute of Technology for their inspiration and valuable guidance during entire duration.

We are extremely thankful to our Artificial Intelligence and Machine Learning faculty in-charge **A.Radhika,** Dept of Computer Science and Engineering, CMR Institute of Technology for his constant guidance, encouragement and moral support throughout the project.

We express our thanks to all staff members and friends for all the help and coordination extended in bringing out this Project successfully in time.

Finally, we are very much thankful to our parents and relatives who guided directly or indirectly for successful completion of the project.

A.Priaanka	**23R01A05CR**
A.Siri	**23R01A05CT**
A.Niranjan	**23R01A05CU**

INDEX

1. INTRODUCTION

Rainfall prediction is a critical aspect of weather forecasting that significantly impacts agriculture, water resource management, urban planning, and disaster preparedness. In recent years, advancements in artificial intelligence and machine learning have revolutionized the field of meteorology, enabling the development of predictive models that can analyse large datasets and identify complex patterns to forecast rainfall accurately.

The objective of this microproject is to develop a predictive model for studying and forecasting the rainfall patterns in our society. By leveraging historical weather data and applying machine learning algorithms, the model aims to provide insights into precipitation trends. Such insights can assist in planning water usage, mitigating flood risks, and supporting sustainable development practices within the community.

This project will involve:

1. **Data Collection and Preprocessing**: Gathering historical rainfall data, along with other relevant weather parameters like temperature, humidity, and wind speed.
2. **Model Development**: Using machine learning techniques such as regression, decision trees, or neural networks to predict rainfall.
3. **Evaluation and Deployment**: Assessing the model's accuracy using appropriate performance metrics and demonstrating its practical applications.

By the end of this microproject, we aim to establish a foundation for using AI and ML tools in solving real-world problems while understanding the complexities of weather prediction. This project will serve as a valuable learning experience for applying theoretical concepts to practical scenario. Rainfall prediction is a critical aspect of weather forecasting that significantly impacts agriculture, water resource management, urban planning, and disaster preparedness. In recent years, advancements in artificial intelligence and machine learning have revolutionized the field of meteorology, enabling the development of predictive models that can analyze .

In addition to predicting rainfall, this project will explore the influence of multiple atmospheric variables and their correlation with precipitation levels. Advanced data visualization tools such as heat maps, scatter plots, and trend graphs will be used to interpret the relationships between parameters like temperature, humidity, air pressure, and wind speed. Understanding these correlations can help identify seasonal behavior, unusual climate changes, and extreme weather conditions. By analysing data trends, the model can be refined over time to improve its prediction accuracy and reliability.

Furthermore, the implementation of this rainfall prediction model can benefit stakeholders such as farmers, local authorities, and environmental agencies. Farmers can plan irrigation schedules and crop selection more effectively, reducing water wastage and increasing agricultural productivity. Urban planners can use the insights to prepare for heavy rainfall events and design better drainage and flood control systems. Overall, this project demonstrates how integrating machine learning into meteorology can contribute to smarter decision-making and support sustainable environmental practices in our society.

2. PROCEDURE

Step 1: Problem Identification

- Objective: Develop a machine learning model to predict rainfall in your society based on historical weather data.
- Importance: Accurate rainfall prediction can assist in better planning for agriculture, water management, and flood prevention.

Step 2: Data Collection

- Source Identification:

  Collect historical weather data from reliable sources such as:

  - Government websites (e.g., India Meteorological Department).
  - Public datasets (e.g., Kaggle, UCI Machine Learning Repository).
  - Local weather station reports.

- Key Variables: Gather data on:

  - Rainfall (target variable).
  - Temperature (daily maximum andminimum).
  - Humidity levels.
  - Wind speed and direction.
  - Atmospheric pressure.
  - Date and time stamps.

- Duration: Collect data for at least the last 5–10 years for accurate trend analysis.

Step 3: Data Preprocessing

Cleaning the Data:

- Remove duplicate or incomplete records. o Handle missing data using techniques like interpolation or imputation (e.g., replace missing values with the mean or median).

- Outlier Removal: Identify outliers in features like rainfall amounts using statistical methods or visualization tools (e.g., box plots) and address them as needed.
- Feature Engineering:
  - Add derived features if required, such as rolling averages of rainfall or temperature. o Convert categorical data (e.g., seasons) into numerical formats using encoding techniques if necessary.
- Normalization: Normalize numerical features to scale them within a range (e.g., 0 to 1) for better performance with machine learning models.
- Data Splitting: Divide the dataset into:
  - Training set (80%): Used to train the model.
  - Testing set (20%): Used to evaluate the model's accuracy.

Step 4: Exploratory Data Analysis (EDA)

Visual Analysis:

- Plot graphs to understand relationships between rainfall and other features (e.g., scatter plots, line charts). o Use correlation heatmaps to identify the features most correlated with rainfall.

- Statistical Analysis:
  - Calculate summary statistics (mean, median, standard deviation) for each feature. o Examine seasonal variations or trends in rainfall patterns.

Step 5: Model Selection

- Choose an appropriate machine learning model for the dataset:
  - Linear Regression: If the relationship between features and rainfall is linear.
  - Decision Trees or Random Forest: To handle non-linear

    relationships and feature interactions. o Support Vector Machines (SVM): For smaller datasets with clear boundaries. o Neural Networks: For larger, more complex datasets with nonlinear dependencies.

Step 6: Model Training

- Train the selected model using the training dataset.
- Techniques:
- Use cross-validation to prevent overfitting.
- Optimize the model parameters (hyperparameter tuning) using methods like Grid Search or Random Search.

Step 7: Model Evaluation

- Evaluate the model's performance using the testing dataset.
- Performance Metrics:
- Mean Absolute Error (MAE): Measures the average magnitude of errors. Mean Squared Error (MSE): Penalizes larger errors more heavily.
- R-Squared (R^2): Indicates the proportion of variance explained by the model.

Step 8: Prediction and Analysis

- Use the trained model to predict rainfall for a specific date, time, or set of weather conditions.
- Analyze the predictions for:
- Accuracy: Compare predicted values with actual rainfall data (if available).
- Insights: Identify trends, such as seasonal rainfall patterns or anomalies.

Step 9: Visualization of Results

- Create clear visualizations to present the model's results and findings:
- Predicted vs. Actual Rainfall: Line or scatter plots to compare outcomes.
- Feature Importance: Bar graphs to show which factors (e.g., humidity, temperature) contributed most to rainfall prediction.

Step 10: Documentation and Conclusion

- Summary of Results: Document the model's accuracy, limitations, and strengths.
- Applications: Highlight how this model can help the society (e.g., flood prevention, farming decisions).
- Limitations and Improvements:
- Discuss any issues faced, such as limited data or model biases.
- Suggest improvements, such as adding more features or using advanced algorithms like deep learning.
- Report Submission: Prepare a detailed report covering the methodology.

3. REQUIREMENTS (Hardware and Software)

Hardware Requirements

1. Personal Computer or Laptop:
    - Processor: Intel i5 or equivalent and above (preferably multicore for faster computations). o RAM: Minimum 8 GB (16 GB recommended for large datasets). o Storage: At least 10 GB free space for data storage and software installation. o GPU (Optional): Dedicated GPU (e.g., NVIDIA) is beneficial for training complex models like neural networks but is not mandatory for simpler models.
2. External Storage (Optional):
    - USB drive or external hard disk to store backups of datasets and project files.
3. Internet Connection:
    - Stable internet connection for downloading datasets, libraries, and tools.

Software Requirements

1. Operating System:
    - Windows 10 or later / macOS / Linux (Ubuntu preferred for opensource environments).
2. Programming Environment:
    - Python 3.x: A versatile programming language widely used for AI and ML projects.

3. Integrated Development Environment (IDE):
    - Jupyter Notebook (recommended for step-by-step execution and visualization). o PyCharm or Visual Studio Code (optional, for advanced coding).
4. Required Python Libraries:
    - Data Handling and Processing:
        - pandas: For data manipulation and preprocessing.
        - numpy: For numerical operations. oData Visualization:
        - matplotlib: For creating visualizations.
        - seaborn: For advanced and visually appealing statistical plots. o Machine Learning:
        - scikit-learn: For implementing ML algorithms such aslinear regression, decision trees, etc. oEvaluation Metrics:
        - statsmodels: For statistical analysis and model evaluation. oOthers (Optional):
        - tensorflow or keras: For neural networks, if required.
5. Dataset Source:
    - Ensure that datasets are in formats like .csv, .xlsx, or .json for easy integration into Python scripts.
6. Version Control (Optional): o Git/GitHub for version control and collaborative development.
7. Documentation Tools:
    - Microsoft Word or Google Docs for preparing the project report. o

Canva or PowerPoint for creating visuals and presentations.

- Optional Tools for Enhancement

1. Cloud Computing Platforms:
    - Google Colab (free and includes GPU support for training models).
    - Kaggle Kernels (for online execution and dataset exploration).
2. Database (Optional):
    - SQLite or MySQL if dealing with large datasets requiring efficient storage and retrieval.

These hardware and software requirements provide the necessary infrastructure to implement the rainfall prediction microproject effectively. Let me know if you need assistance with setting up any of these tools!

RESULT

Fig-1: Now let's load the dataset into the panda's data frame and print its first five rows.

```
df = pd.read_csv('Rainfall.csv')
df.head()
```

Output:

	day	pressure	maxtemp	temparature	mintemp	dewpoint	humidity	cloud	rainfall	sunshine	winddirection	windspeed
0	1	1025.9	19.9	18.3	16.8	13.1	72	49	yes	9.3	80.0	26.3
1	2	1022.0	21.7	18.9	17.2	15.6	81	83	yes	0.6	50.0	15.3
2	3	1019.7	20.3	19.3	18.0	18.4	95	91	yes	0.0	40.0	14.2
3	4	1018.9	22.3	20.6	19.1	18.8	90	88	yes	1.0	50.0	16.9
4	5	1015.9	21.3	20.7	20.2	19.9	95	81	yes	0.0	40.0	13.7

First Five rows of the dataset

Fig-2: Now let's check the size of the dataset.

Python

```
df.shape
```

Output:

```
(366, 12)
```

Fig-3: Let's check which column of the dataset contains which type of data.

```python
df.info()
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 366 entries, 0 to 365
Data columns (total 12 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   day                    366 non-null    int64
 1   pressure               366 non-null    float64
 2   maxtemp                366 non-null    float64
 3   temparature            366 non-null    float64
 4   mintemp                366 non-null    float64
 5   dewpoint               366 non-null    float64
 6   humidity               366 non-null    int64
 7   cloud                  366 non-null    int64
 8   rainfall               366 non-null    object
 9   sunshine               366 non-null    float64
 10          winddirection  365 non-null    float64
 11  windspeed              365 non-null    float64
dtypes: float64(8), int64(3), object(1)
memory usage: 34.4+ KB
```

Fig-4: As per the above information regarding the data in each column, we can observe that there are no null values.

```python
df.describe().T
```

Output:

	count	mean	std	min	25%	50%	75%	max
day	366.0	15.756831	8.823592	1.0	8.000	16.00	23.000	31.0
pressure	366.0	1013.742623	6.414776	998.5	1008.500	1013.00	1018.100	1034.6
maxtemp	366.0	26.191257	5.978343	7.1	21.200	27.75	31.200	36.3
temparature	366.0	23.747268	5.632813	4.9	18.825	25.45	28.600	32.4
mintemp	366.0	21.894536	5.594153	3.1	17.125	23.70	26.575	30.0
dewpoint	366.0	19.989071	5.997021	-0.4	16.125	21.95	25.000	26.7
humidity	366.0	80.177596	10.062470	36.0	75.000	80.50	87.000	98.0
cloud	366.0	71.128415	21.798012	0.0	58.000	80.00	88.000	100.0
sunshine	366.0	4.419399	3.934398	0.0	0.500	3.50	8.200	12.1
winddirection	365.0	101.506849	81.723724	10.0	40.000	70.00	190.000	350.0
windspeed	365.0	21.536986	10.069712	4.4	13.700	20.50	27.900	59.5

Fig-5: The data which is obtained from the primary sources is termed the raw data and required a lot of preprocessing before we can derive any conclusions from it or do some modeling on it. Those preprocessing steps are known as data cleaning and it includes, outliers removal, null value imputation, and removing discrepancies of any sort in the data inputs.

Python

```
Python

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

Output:

```
day                0
pressure           0
maxtemp            0
temparature        0
mintemp            0
dewpoint           0
humidity           0
cloud              0
rainfall           0
sunshine           0
winddirection      1
windspeed          1
dtype: int64
```

Fig-6: So there is one null value in the 'winddirection' as well as the 'windspeed' column. But what's up with the column name wind direction?

Python

```
df.columns
```

Output:

```
Index(['day', 'pressure ', 'maxtemp', 'temperature', 'mintemp', 'dewpoint',
       'humidity ', 'cloud ', 'rainfall', 'sunshine', '         winddirection',
       'windspeed'],
      dtype='object')
```

Fig-7: Here we can observe that there are unnecessary spaces in the names of the columns let's remove that.

Python

```
df.rename(str.strip,
          axis='columns',
          inplace=True)

df.columns
```

Output:

```
Index(['day', 'pressure', 'maxtemp', 'temperature', 'mintemp', 'dewpoint',
       'humidity', 'cloud', 'rainfall', 'sunshine', 'winddirection',
       'windspeed'],
      dtype='object')
```

Fig-8: Now it's time for null value imputation.

Python

```
for col in df.columns:

  # Checking if the column contains
  # any null values
  if df[col].isnull().sum() > 0:
    val = df[col].mean()
    df[col] = df[col].fillna(val)

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

Output:

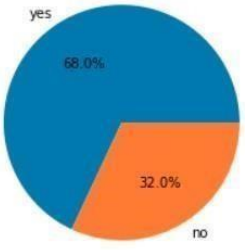
```
0
```

Fig-9: EDA is an approach to analyzing the data using visual techniques. It is used to discover trends, and patterns, or to check assumptions with the help of statistical summaries and graphical representations. Here we will see how to check the [data imbalance] and skewness of the data.

Python

```
plt.pie(df['rainfall'].value_counts().values,
        labels = df['rainfall'].value_counts().index,
        autopct='%1.1f%%')
plt.show()
```

Output:



Pie chart for the number of data for each target

Fig-10: The observations we have drawn from the above dataset are very much similar to what is observed in real life as well.

Python

```
features = list(df.select_dtypes(include = np.number).columns)
features.remove('day')
print(features)
```

Output:

```
['pressure', 'maxtemp', 'temperature', 'mintemp', 'dewpoint', 'humidity', 'cloud',
'sunshine', 'winddirection', 'windspeed']
```

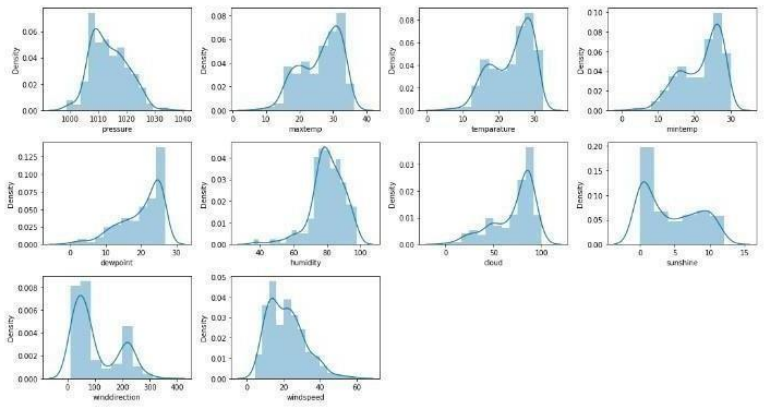
Fig-11: Let's check the distribution of the continuous features given in the dataset.

Python

```
plt.subplots(figsize=(15,8))

for i, col in enumerate(features):
  plt.subplot(3,4, i + 1)
  sb.distplot(df[col])
plt.tight_layout()
plt.show()
```

Output:



Distribution plot for the columns with continuous data

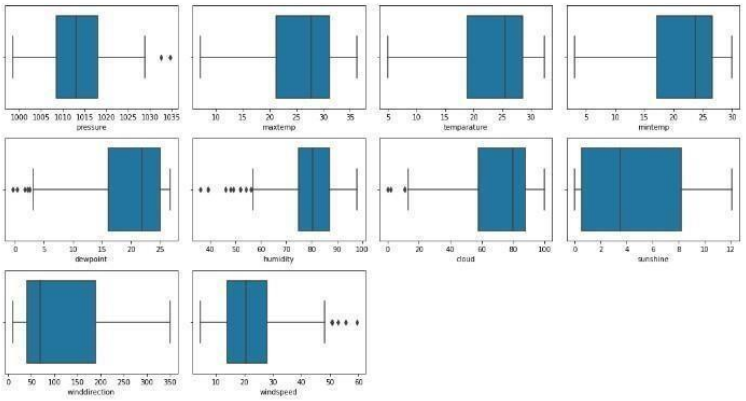
Fig-12: Let's draw boxplots for the continuous variable to detect the outliers present in the data.

Python

```
plt.subplots(figsize=(15,8))

for i, col in enumerate(features):
  plt.subplot(3,4, i + 1)
  sb.boxplot(df[col])
plt.tight_layout()
plt.show()
```

Output:



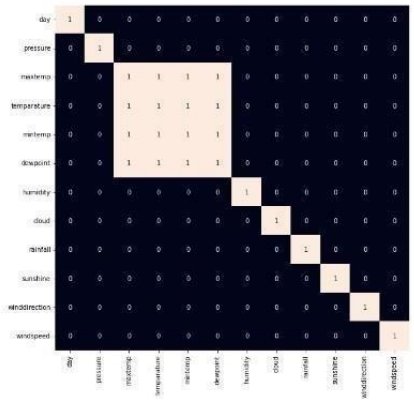
Box plots for the columns with continuous data

Fig-13: Sometimes there are highly correlated features that just increase the dimensionality of the feature space and do not good for the model's performance. So we must check whether there are highly correlated features in this dataset or not.

Python

```
plt.figure(figsize=(10,10))
sb.heatmap(df.corr() > 0.8,
           annot=True,
           cbar=False)
plt.show()
```

Output:



Heat map to detect highly correlated features

Fig-14: Now let's train some state-of-the-art models for classification and train them on our training data.

- LogisticRegression
- XGBClassifier SVC

```
models = [LogisticRegression(), XGBClassifier(), SVC(kernel='rbf',
probability=True)]

for i in range(3):
  models[i].fit(X, Y)

  print(f'{models[i]} : ')

  train_preds = models[i].predict_proba(X)
  print('Training Accuracy : ', metrics.roc_auc_score(Y, train_preds[:,1]))

  val_preds = models[i].predict_proba(X_val)
  print('Validation Accuracy : ', metrics.roc_auc_score(Y_val, val_preds[:,1]))
  print()
```

Output:

```
LogisticRegression() :
Training Accuracy :  0.8893967324057472
Validation Accuracy :  0.8966666666666667

XGBClassifier() :
Training Accuracy :  0.9903285270573975
Validation Accuracy :  0.8408333333333333

SVC(probability=True) :
Training Accuracy :  0.9026413474407211
Validation Accuracy :  0.8858333333333333
```

Fig-15: Let's plot the [confusion matrix] as well for the validation data using the SVC model.

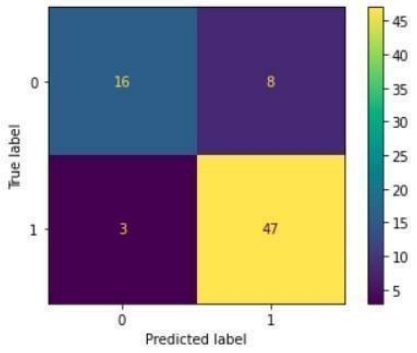
Python

```python
import matplotlib.pyplot as plt
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn import metrics

ConfusionMatrixDisplay.from_estimator(models[2], X_val, Y_val)
plt.show()

# This code is modified by Susobhan Akhuli
```

Output:



Confusion matrix for the validation data

Fig-16: Let's plot the classification report as well for the validation data using the SVC mode.

Python

```
print(metrics.classification_report(Y_val,
                                    models[2].predict(X_val)))
```

Output:

```
              precision    recall  f1-score   support

           0       0.84      0.67      0.74        24
           1       0.85      0.94      0.90        50

    accuracy                           0.85        74
   macro avg       0.85      0.80      0.82        74
weighted avg       0.85      0.85      0.85        74
```

4. CONCLUSION

The successful completion of this microproject demonstrates the practical application of artificial intelligence and machine learning techniques to solve real-world problems, such as predicting rainfall for a specific locality. By utilizing historical weather data and leveraging machine learning algorithms, we developed a predictive model capable of analyzing complex patterns and providing rainfall forecasts with reasonable accuracy.

Through this project, we gained hands-on experience in data preprocessing, feature selection, model training, and evaluation. The insights derived from the model can assist in better decision-making for activities such as agricultural planning, water resource management, and disaster preparedness in our society.

Despite its success, the model has certain limitations, such as dependency on the quality and quantity of the input data. Additionally, external factors like sudden climatic changes and unrecorded variables may impact its predictions. Future improvements can include using larger datasets, integrating advanced models like deep learning, and incorporating real-time weather data for more accurate and dynamic predictions.

This microproject underscores the transformative potential of AI and ML in addressing community-specific challenges, laying the foundation for future advancements in weather forecasting and related fields.

5. REFERENCES

1. https://towardsdatascience.com/machine-learning-for-weather-forecasting-11b74d467d1d
2. https://www.geeksforgeeks.org/weather-forecasting-using-machinelearningin-python/
3. https://analyticsindiamag.com/using-neural-networks-for-rainfallpredictiona- step-by-step-guide/
4. https://scikit-learn.org/stable/