Project Design Phase-II

Technology Stack (Architecture & Stack)

Date	20 November 2023
Team ID	591569
Project Name	Machine Learning Approach for Predicting the Rainfall
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

Example: Rainfall Prediction using an Ensemble Machine Learning Model

Reference: https://www.mdpi.com/2071-1050/15/7/5889

Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services
5. Indicate interface to machine learning models (if applicable)

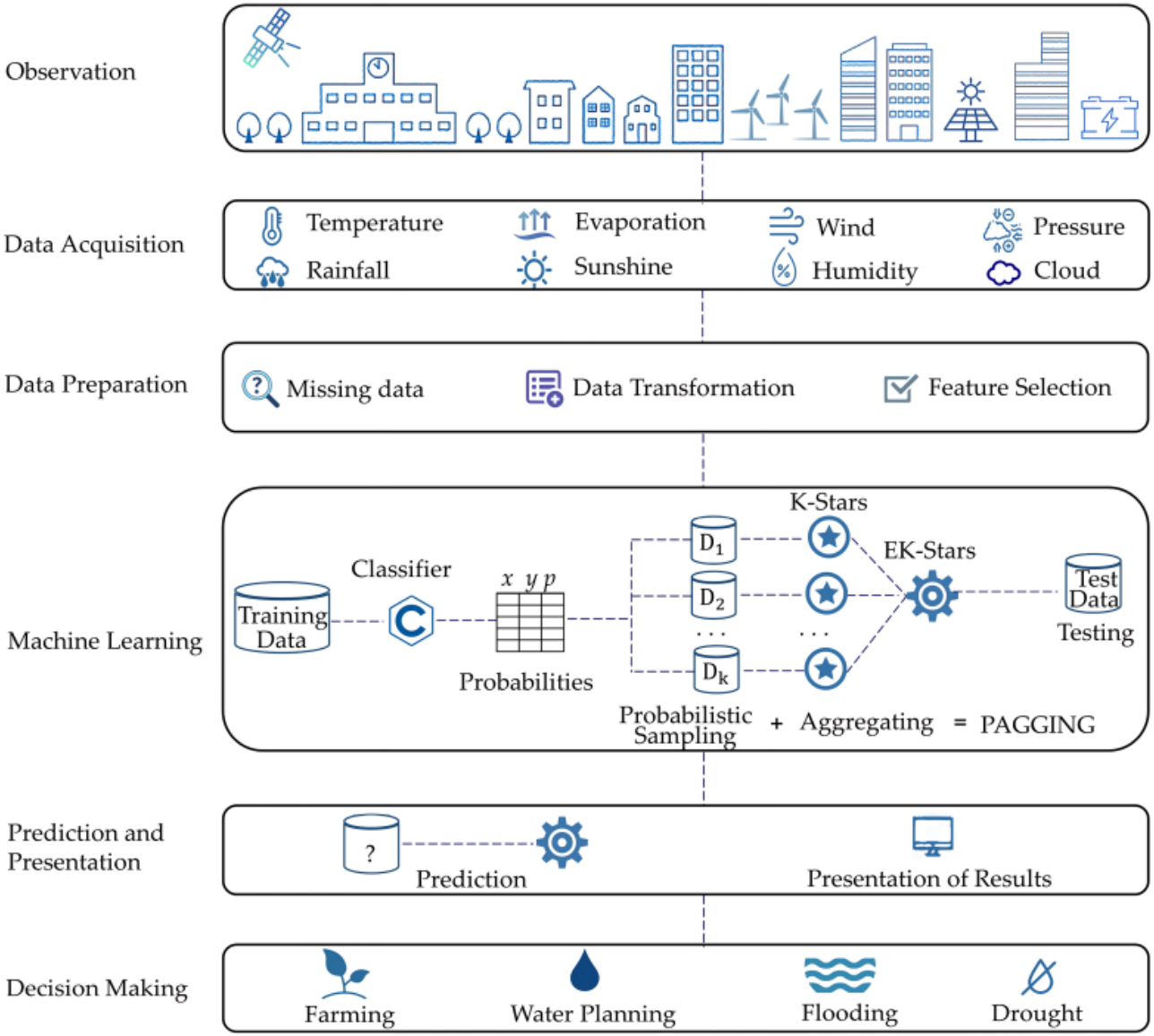
Observation
Data Acquisition
Temperature
Evaporation
Wind
Pressure
Rainfall
Sunshine
Humidity
Cloud
Data Preparation
Missing data
Data Transformation
Feature Selection
Machine Learning
K-Stars
Classifier
EK-Stars
Training Data
x y p
Test Data
Testing
Probabilities
Probabilistic Sampling
+ Aggregating = PAGGING
Prediction and Presentation
Prediction
Presentation of Results
Decision Making
Farming
Water Planning
Flooding
Drought

Table-1: Components & Technologies:

S.No	Component	Description	Technology
1.	Data Collection	Responsible for acquiring historical rainfall data from various sources.	Web scraping tools, APIs for meteorological data, data connectors.
2.	Data Preprocessing and Feature Engineering	Cleans and processes the data, extracts relevant features, and engineers new ones.	Python libraries such as Pandas, NumPy for preprocessing, and feature engineering.
3.	Machine Learning Model Development	Develops and trains machine learning models on historical data.	Python-based machine learning libraries like scikit-learn, TensorFlow, or PyTorch.
4.	Real-time Prediction Module	Handles real-time predictions using the trained machine learning model.	Streaming platforms (e.g., Apache Kafka), microservices architecture with containers (e.g., Docker, Kubernetes).
5.	Scalable Infrastructure	Provides a scalable and flexible computing environment for model training and real-time predictions.	Cloud platforms such as AWS, Azure, or Google Cloud.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	API Integration	Exposes APIs for external systems to query real-time and forecasted rainfall data.	RESTful APIs, JSON for data exchange, API Gateway for managing API access.
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
9.	User Interface/Dashboard	Presents historical data and real-time predictions in an interactive and user-friendly format.	Web-based dashboard using frontend frameworks like React or Angular.

10.	Monitoring and Logging	Monitors system health, logs events, and performance metrics.	Logging frameworks (e.g., ELK Stack - Elasticsearch, Logstash, Kibana), monitoring tools.
11.	Security Measures	Implements security measures to protect data and system integrity.	Encryption algorithms for data at rest (e.g., AES), HTTPS for secure communication, role-based access control (RBAC).

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Real-time Prediction and Forecasting	The system provides real-time rainfall predictions and forecasts for future time periods, allowing users to make informed decisions based on the latest weather information.	Streaming platforms (e.g., Apache Kafka) for handling live data streams, machine learning models for forecasting.
2.	Historical Data Visualization	Users can explore and visualize historical rainfall patterns through an intuitive and interactive dashboard, gaining insights into past weather trends.	Web-based dashboard using frontend frameworks (e.g., React, Angular, Vue.js), backend APIs for data retrieval.
3.	Scalability and Flexibility	The system is designed to scale with increasing data volumes and user demands, providing flexibility to adapt to changing requirements.	Cloud platforms (e.g., AWS, Azure) for scalable and flexible computing resources, containerization (e.g., Docker, Kubernetes) for resource management.
4.	User-Friendly Interface	The application features an intuitive and user-friendly interface, allowing users to	Responsive web design, frontend frameworks (e.g., React, Angular,

		easily navigate and interpret rainfall predictions and historical data.	Vue.js), interactive data visualization libraries.
5.	Security Measures	The application prioritizes data security, implementing encryption for data at rest and in transit, secure APIs, and access controls.	Encryption algorithms (e.g., AES), HTTPS for secure communication, API security mechanisms, role-based access control (RBAC).

References:

https://www.mdpi.com/2071-1050/15/7/5889

https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00545-4

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9099780/