Weather Forecasting System Report

1. Introduction

Accurate weather forecasting is essential for farmers to make informed decisions about irrigation, planting, and harvesting. This report outlines the development of a machine learning model to predict rainfall based on historical weather data. Additionally, it describes the system design for real-time weather predictions using IoT sensors.

2. Data Preprocessing (Conducted in RStudio)

2.1 Understanding the Dataset

The dataset consists of 311 daily weather observations, including the following features:

- **Date** – The day of observation.
- **Average Temperature (°C)** – The mean daily temperature.
- **Humidity (%)** – The percentage of moisture in the air.
- **Average Wind Speed (km/h)** – The speed of the wind.
- **Cloud Cover (%)** – The fraction of the sky covered by clouds.
- **Pressure (hPa)** – Atmospheric pressure.
- **Rain or Not (Binary: 1=Rain, 0=No Rain)** – The target variable.

2.2 Data Cleaning & Handling Missing Values in R

```
install.packages(c("dplyr", "ggplot2", "GGally"))
library(dplyr)
library(ggplot2)
library(GGally)

weather_data <- read.csv("weather_data.csv")

weather_data$date <- as.Date(weather_data$date, format="%Y-%m-%d")

print(colSums(is.na(weather_data)))

weather_data <- weather_data %>% mutate(
  avg_temperature = ifelse(is.na(avg_temperature), mean(avg_temperature, na.rm = TRUE),
avg_temperature),
  humidity = ifelse(is.na(humidity), mean(humidity, na.rm = TRUE), humidity),
```

```
  avg_wind_speed = ifelse(is.na(avg_wind_speed), mean(avg_wind_speed, na.rm = TRUE), avg_wind_speed),
  cloud_cover = ifelse(is.na(cloud_cover), mean(cloud_cover, na.rm = TRUE), cloud_cover)
)

weather_data$rain_or_not <- as.factor(weather_data$rain_or_not)
```

3. Exploratory Data Analysis (EDA) in RStudio

3.1 Feature Distributions

```
ggplot(weather_data, aes(x=avg_temperature)) +
  geom_histogram(binwidth=2, fill="skyblue", color="black") +
  ggtitle("Temperature Distribution")
```

3.2 Correlation Analysis

```
ggpairs(weather_data, columns = c("avg_temperature", "humidity", "avg_wind_speed", "cloud_cover", "pressure"))
```

4. Machine Learning Model Training (RStudio)

4.1 Splitting Data into Training & Test Sets

```
library(caret)

set.seed(42)
trainIndex <- createDataPartition(weather_data$rain_or_not, p=0.8, list=FALSE)
trainData <- weather_data[trainIndex,]
testData <- weather_data[-trainIndex,]
```

4.2 Training Machine Learning Models

```
log_model <- train(rain_or_not ~ ., data=trainData, method="glm", family=binomial)

tree_model <- train(rain_or_not ~ ., data=trainData, method="rpart")

rf_model <- train(rain_or_not ~ ., data=trainData, method="rf")
```

4.3 Model Evaluation

```
rf_pred <- predict(rf_model, testData)
confusionMatrix(rf_pred, testData$rain_or_not)
```

4.4 Predicting the Next 21 Days

```
future_predictions <- predict(rf_model, newdata=testData[1:21,])
print(future_predictions)
```

5. System Diagrams for Real-Time Predictions

5.1 Steps to Create System Diagrams in R

Install & Load DiagrammeR Package:

```
install.packages("DiagrammeR")
library(DiagrammeR)
```

Create a Flowchart for the System Design:

```
DiagrammeR("graph TD;
  A[IoT Sensors] -->|Raw Data| B[Data Processing];
  B -->|Stored Data| C[Database & Storage];
  C -->|Training & Prediction| D[Machine Learning Model];
  D -->|Predictions| E[API Layer];
  E -->|Visualization| F[Frontend Dashboard];")
```

Export the Diagram as an Image:

```
install.packages("DiagrammeRsvg")
install.packages("rsvg")
library(DiagrammeRsvg)
library(rsvg)

svg_code <- grViz("graph TD;
  A[IoT Sensors] -->|Raw Data| B[Data Processing];
  B -->|Stored Data| C[Database & Storage];
  C -->|Training & Prediction| D[Machine Learning Model];
  D -->|Predictions| E[API Layer];
  E -->|Visualization| F[Frontend Dashboard];") %>% export_svg()

writeLines(svg_code, "mlops_diagram.svg")
```

```
rsvg_png("mlops_diagram.svg", "mlops_diagram.png")
```