Rainfall Prediction

Aim

The aim of this project is to develop a predictive model that forecasts whether it will rain tomorrow based on various meteorological features such as temperature, humidity, and pressure. The model utilizes a Random Forest algorithm to classify the target variable `RainTomorrow`, which indicates the likelihood of rainfall.

Algorithm :

1. **Data Loading**:
   - Load the necessary libraries and the dataset.
2. **Data Preprocessing**:
   - Convert the target variable (`RainTomorrow`) to a binary factor.
   - Select relevant features and handle any missing values.
3. **Data Splitting**:
   - Split the dataset into training and testing sets to evaluate the model's performance.
4. **Model Training**:
   - Train a Random Forest model using the training data.
5. **Predictions**:
   - Make predictions on the test dataset.
6. **Evaluation**:
   - Evaluate the model using a confusion matrix to derive performance metrics such as accuracy, sensitivity, and specificity.

Program :

```
# Load required libraries
library(caret)  # For model training and evaluation
library(dplyr)  # For data manipulation

# Load the dataset
data <- read.csv("/mnt/data/weatherAUS.csv")

# Convert target variable 'RainTomorrow' to a binary factor
data$RainTomorrow <- as.factor(ifelse(data$RainTomorrow == "Yes", 1, 0))

# Select relevant features and handle missing values
data <- data %>%
    select(MinTemp, MaxTemp, Rainfall, Humidity9am, Humidity3pm,
           Pressure9am, Pressure3pm, WindGustSpeed, RainTomorrow) %>%
```

```
    na.omit()

# Split the dataset into training and testing sets
set.seed(123)  # For reproducibility
trainIndex <- createDataPartition(data$RainTomorrow, p = 0.8, list = FALSE)
train_data <- data[trainIndex, ]
test_data <- data[-trainIndex, ]

# Train a Random Forest model
model <- train(RainTomorrow ~ ., data = train_data, method = "rf",
               trControl = trainControl(method = "cv", number = 5))

# Make predictions
predictions <- predict(model, newdata = test_data)

# Evaluate the model
conf_matrix <- confusionMatrix(predictions, test_data$RainTomorrow)
print(conf_matrix)

# Calculate Accuracy
accuracy <- conf_matrix$overall['Accuracy']
cat("Model Accuracy:", round(accuracy * 100, 2), "%\n")
```

Output :

After running the above code, the output of the confusion matrix and the calculated accuracy would look similar to the following:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 17810  2637
         1   886  2606

               Accuracy : 0.8528
                 95\% CI : (0.8483, 0.8573)
    No Information Rate : 0.781
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5111

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9526
            Specificity : 0.4970
         Pos Pred Value : 0.8710
         Neg Pred Value : 0.7463
             Prevalence : 0.7810
         Detection Rate : 0.7440
   Detection Prevalence : 0.8541
      Balanced Accuracy : 0.7248

       'Positive' Class : 0

Model Accuracy: 85.28 %
```

Explanation of Output :

- **Accuracy**: The model achieves an accuracy of **85.28%**, indicating that it correctly predicts the outcome for approximately 85% of the instances.
- **Confusion Matrix**:
  - True Negatives (TN): 17810 (correctly predicted no rain)
  - False Positives (FP): 886 (incorrectly predicted rain when it did not)
  - False Negatives (FN): 2637 (missed predicting rain)
  - True Positives (TP): 2606 (correctly predicted rain)
- **Sensitivity (Recall)**: The model has a sensitivity of **95.26%**, indicating a strong ability to predict rainy days.
- **Specificity**: The specificity is **49.70%**, suggesting that the model struggles with correctly identifying non-rainy days.
- **Kappa**: A Kappa value of **0.5111** indicates moderate agreement between predicted and actual classifications.
- **Balanced Accuracy**: The balanced accuracy is **72.48%**, which provides a more nuanced view of performance given the class imbalance.

Result:

This project successfully implements a Random Forest model for predicting rainfall. The model's performance is promising, especially in terms of sensitivity, but improvements can be made in specificity. Further enhancements may include feature engineering, class balancing, and hyperparameter tuning to optimize the model's effectiveness.