Faculty of Computing and Information Technology

University of the Punjab,

Lahore

Artificial Intelligence Lab 10

Instructor: Qamar U Zaman

1. Random Forest Algorithm

Objective:
The goal of this lab is to understand the Random Forest algorithm.

Key Concepts:

1. **Random Forest**:
   Random Forest is an ensemble learning method that constructs multiple decision trees and combines their outputs to improve classification accuracy.
2. **Entropy**:
   Entropy is a measure of uncertainty or impurity in a dataset. Lower entropy means the data is more homogenous. In Random Forest, entropy is used to select the attribute that best splits the dataset at each decision node.
3. **Information Gain**:
   Information Gain measures the effectiveness of an attribute in classifying the dataset. It's used to determine the best split for a node in a decision tree.
4. **Random Forest Construction**:
   Random Forest involves the following steps:
   - Generate multiple decision trees by selecting random subsets of the dataset and features.
   - Use entropy and information gain to split the dataset at each node in each tree.
   - Combine the outputs (classifications) of all trees to make a final decision.

Problem Description:

Given a simple dataset of weather conditions and whether to play a game, your task is to build a **Random Forest** classifier that can predict the outcome (Play? Yes/No) based on the weather and temperature conditions.

Dataset:

Weather	Temperature	Play?
Sunny	Hot	No
Overcast	Hot	Yes
Rainy	Mild	Yes
Sunny	Mild	No
Overcast	Mild	Yes
Rainy	Hot	No

Steps:

1. **Calculate Entropy**:
   For both the entire dataset and individual attributes.
2. **Calculate Information Gain**:
   For each attribute (Weather and Temperature) to determine which attribute should be used for splitting.
3. **Build Random Forest**:
   - Construct multiple decision trees (in this case, we will use just 2 for simplicity).
   - Use **entropy** as the criterion for selecting splits in each tree.
4. **Classify Data**:
   Use the built Random Forest model to predict the outcome for new data points.

Code Template:

```
# Calculate entropy of a dataset
def calculate_entropy(data, target_col):
    """
    Calculate the entropy of the target column in the dataset.
    """
    pass

# Calculate information gain for an attribute
def calculate_information_gain(data, attribute, target_col):
    """
    Calculate the information gain for a given attribute in the dataset.
    """
    pass

# Build the decision tree
def build_tree(data, attributes, target_col, depth=0, max_depth=3):
    """
    Recursively build the decision tree using entropy and information gain.
    """
    pass

# Predict the class for a given data point
def predict(tree, data_point):
    """
    Traverse the tree to predict the class for a given data point.
    """
    pass

# Build Random Forest
def build_random_forest(data, attributes, target_col, n_trees=2):
    """
    Build a Random Forest by generating multiple decision trees.
    """
    pass
```

Task Description:

Students are expected to:

1. **Test with different datasets**:
   - Use the provided sample dataset or create your own datasets with different conditions and outcomes.
2. **Analyze the decision tree structure for various splits**:
   - Examine how the decision tree is built using entropy and information gain.
3. **Experiment with new test data points and verify predictions**:
   - Use the Random Forest model to predict new data points and evaluate the predictions.

2. Linear Regression

Objective:
The objective of this lab is to implement a **Simple Linear Regression** model from scratch, calculate the **slope** and **intercept** using basic formulas, and evaluate the model using **Mean Squared Error (MSE)**.

Key Concepts:

1. **Linear Regression**:
   Linear Regression models the relationship between an independent variable X and a dependent variable Y.
2. **Mean Squared Error (MSE)**:
   MSE is used to evaluate the model's performance. It calculates the average squared difference between the predicted values Y^ and the actual values Y.

Problem Description:

You are given a simple dataset with one feature X (independent variable) and one target Y (dependent variable). Your task is to implement **Simple Linear Regression**, calculate the **slope** and **intercept** manually, and evaluate the model using **Mean Squared Error (MSE)**.

Dataset:

X	Y
1	2
2	4
3	5
4	7
5	8

Steps:

1. **Calculate the Mean of X and Y.**
2. **Calculate the Slope.**
3. **Calculate the Intercept.**
4. **Make Predictions**.
5. **Calculate the Mean Squared Error (MSE)**.

Code Template:

```
# Calculate the mean of a list of values
def calculate_mean(values):

        pass

# Calculate the slope (theta_1)
def calculate_slope(X, Y, mean_X, mean_Y):

        pass


# Calculate the intercept (theta_0)
def calculate_intercept(mean_X, mean_Y, slope):

        pass


# Make predictions using the learned model
def predict(X, theta_0, theta_1):

        pass


# Calculate the Mean Squared Error (MSE)
def calculate_mse(Y, Y_pred):

        pass

# Fit the linear regression model
def fit_linear_regression(X, Y):

    # Calculate the slope (theta_1)


    # Calculate the intercept (theta_0)
```

Task Description:

1. **Test with the provided dataset**:
    - Use the provided dataset to calculate the model parameters and make predictions. Then calculate the **Mean Squared Error (MSE)** to evaluate the model's performance.
2. **Experiment with other datasets**:
    - Try using different datasets to test the model and evaluate its performance.
3. **Evaluate the model**:
    - After fitting the model, calculate the **Mean Squared Error (MSE)** for the predictions and compare it to the true values.