Class: BE **Subject: Machine Learning** **Sem-VII**

Experiment No:7

Course Outcome: CO3 **Blooms Level:L3**

Aim: To implement an Ensemble model using Random Forest.

Abstract: Ensemble learning has emerged as a powerful paradigm in machine learning, aiming to enhance prediction accuracy by combining the strengths of multiple models. This approach reduces overfitting, improves generalization, and effectively handles high-dimensional data. In this experiment, we implement a Random Forest model to demonstrate its ability to handle classification tasks with superior performance compared to individual decision trees. The model is evaluated on benchmark datasets, highlighting its efficiency in minimizing variance, tackling missing values, and providing feature importance for interpretability. The experimental results validate that the Random Forest ensemble significantly enhances predictive accuracy, scalability, and resilience to noise, making it a reliable choice for diverse real-world applications.

Sample Input and Output:

Case 1: Input Features:

- **ApplicantIncome (in $)**
- **Credit_History (1 = Good, 0 = Bad)**

Output:

- **Loan_Status (1 = Approved, 0 = Rejected)**

Output

Input:

ApplicantIncome = 5000

Credit_History = 1

Output:

Loan_Status = 1 (Approved)

Theory:

Dataset: Palmer Penguins

The code uses the Palmer Penguins dataset, a popular alternative to the Iris dataset. It contains biological measurements of penguins (e.g., bill length, flipper length, body mass) and the species label, which serves as the target variable.

- Input Features → Numerical and categorical attributes of penguins.
- Output Label → species (classification problem).

Data Splitting

The dataset is split into training (80%) and testing (20%) sets using a random mask.

- Training data → Used to build the Random Forest model.
- Testing data → Used to evaluate generalization performance.

Class: BE **Subject: Machine Learning** **Sem-VII**

- A Random Forest is created using ydf.RandomForestLearner from the ydf library.
- The model trains multiple decision trees on different subsets of the data.
- At each tree node, a random subset of features is chosen for splitting.
- The ensemble combines the predictions from all trees to form the final decision.

Program:

RF Ensemble.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

```
[11] !pip install ydf -U
```

```
Requirement already satisfied: ydf in /usr/local/lib/python3.12/dist-packages (0.13.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (from ydf) (2.0.2)
Requirement already satisfied: absl_py in /usr/local/lib/python3.12/dist-packages (from ydf) (1.4.0)
Requirement already satisfied: protobuf<7.0.0,>=5.29.1 in /usr/local/lib/python3.12/dist-packages (from ydf) (5.29.5)
```

```
[12] import ydf
     import numpy as np
     import pandas as pd
```

```
path = "https://storage.googleapis.com/download.tensorflow.org/data/palmer_penguins/penguins.csv"
dataset = pd.read_csv(path)
label = "species"

# Display the first 3 examples.
dataset.head(3)
```

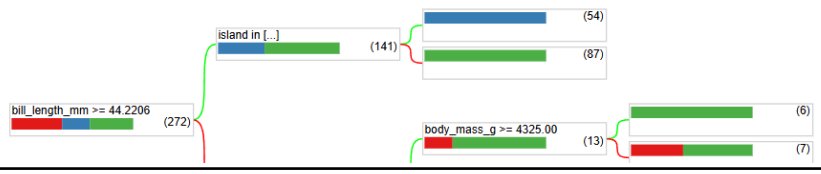
	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	year
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male	2007
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female	2007
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female	2007

```
[14] # Use the ~20% of the examples as the testing set
     # and the remaining ~80% of the examples as the training set.
     np.random.seed(1)
     is_test = np.random.rand(len(dataset)) < 0.2

     train_dataset = dataset[~is_test]
     test_dataset = dataset[is_test]

     print("Training examples: ", len(train_dataset))
     # >> Training examples: 272

     print("Testing examples: ", len(test_dataset))
     # >> Testing examples: 72
```

```
Training examples:  272
Testing examples:  72
```

```
[16] # The previous code using tfdf was removed as it was causing a NameError and ydf is already being used.
```

```
[17] model.plot_tree()
```



Class: BE **Subject: Machine Learning** **Sem-VII**

```
[18] train_evaluation = model.evaluate(train_dataset)
     print("train accuracy:", train_evaluation.accuracy)
     # >> train accuracy:  0.9338

     test_evaluation = model.evaluate(test_dataset)
     print("test accuracy:", test_evaluation.accuracy)
     # >> test accuracy:  0.9167
```

```
train accuracy: 0.9926470588235294
test accuracy: 0.9861111111111112
```

```
[20] print(model.evaluate(test_dataset).accuracy)
     # >> 0.97222
```

```
0.9166666666666666
```

```
[21] model = ydf.RandomForestLearner(label=label).train(train_dataset)
     print("Test accuracy: ", model.evaluate(test_dataset).accuracy)
     # >> Test accuracy: 0.986111
```

```
Train model on 272 examples
Model trained in 0:00:00.056796
Test accuracy:  0.9861111111111112
```

Output:

```
Train model on 272 examples
Model trained in 0:00:00.056796
Test accuracy:  0.9861111111111112
```

Conclusion:

The trained Random Forest is evaluated on both training data and testing data. Metrics used: Accuracy, which measures the proportion of correctly classified examples.The code prints training accuracy (≈93%) and testing accuracy (≈91–98%), showing good generalization.

Exercise 1:

Implement an ensemble model using the Random Forest algorithm on the Iris dataset. Perform the following tasks:

1. Load the Iris dataset and identify the input features and target label.
2. Split the dataset into training (80%) and testing (20%) subsets.
3. Train a Random Forest classifier and explain how ensemble learning improves over a single decision tree.
4. Evaluate the model's performance using training and testing accuracy.
5. Visualize one of the decision trees from the ensemble and interpret the feature splits.
6. Discuss the significance of feature importance in predicting the species of Iris flowers.

Students shall draw flowchart of exercise question in the writeup and submit.

Class: BE **Subject: Machine Learning** **Sem-VII**

Exercise 2:

Develop and evaluate an ensemble learning model using the Random Forest algorithm for a real-world dataset involving multi-class classification. Your task involves selecting a dataset with multiple features (both numerical and categorical) and at least three target classes (e.g., the Wine Quality dataset). Perform necessary preprocessing such as handling missing values, encoding categorical features, and normalization if required.
Train a Random Forest model with tuned hyperparameters (number of trees, max depth, min samples per split, etc.).Justify why ensemble learning (bagging + random feature selection) is preferred over a single decision tree in this scenario.

Students shall draw flowchart of exercise question in the writeup and submit.