Name: Girish Nandanwar	Course: 22CT744 – Lab. Machine Learning
Roll No: A-56	Department: Computer Technology

Practical No.9

Aim: Introduction to TensorFlow and Keras

Theory:

1. Basic Concept

- The tree consists of:
  - Root Node: Represents the entire dataset and the first feature chosen for splitting.
  - Decision Nodes: Intermediate nodes where data is split based on certain feature thresholds.
  - Leaf Nodes: Represent the final class labels or output values.

The goal is to create branches that lead to the most homogeneous subsets possible.

---

2. How It Works

1. The algorithm evaluates all available features and selects the one that best separates the data.
2. It uses measures like Information Gain (Entropy-based) or Gini Index to decide the best split.
3. This process continues recursively until:
   - The data is perfectly classified, or
   - A stopping condition is reached (e.g., max depth or min samples per leaf).

---

3. Splitting Criteria

- Entropy: Measures impurity. Lower entropy = higher purity.

  Entropy=−∑pilog2(pi)Entropy = -\sum p_i \log_2(p_i)Entropy=−∑pilog2(pi)

- Information Gain: Reduction in entropy after a dataset is split on an attribute.

  IG=Entropyparent−∑ninEntropychildiIG = Entropy_{parent} - \sum \frac{n_i}{n} Entropy_{child_i}IG=Entropyparent−∑nniEntropychildi

- Gini Index: Probability of misclassifying a sample.

  Gini=1−∑(pi)2Gini = 1 - \sum (p_i)^2Gini=1−∑(pi)2

---

4. Advantages

- Easy to understand and interpret.
- Requires little data preprocessing (no need for feature scaling).
- Can handle both categorical and numerical data.

---

5. Disadvantages

- Prone to overfitting, especially with small datasets.
- Small variations in data can result in a different tree structure.
- Biased toward features with more levels.

---

6. Evaluation Metrics

To assess performance:

- Accuracy: Ratio of correctly predicted observations to total observations.

- Confusion Matrix: Summarizes true positives, false positives, etc.
- Precision, Recall, F1-Score: For class-wise performance analysis.

Code:

⍰ Import Libraries:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
```

⍰ Load Dataset:

```
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()
```

⍰ Preprocess Data:

```
x_train = x_train / 255.0
x_test = x_test / 255.0
```

⍰ Build Model:

```
model = keras.Sequential([
    layers.Flatten(input_shape=(28, 28)),
    layers.Dense(128, activation='relu'),
    layers.Dense(10, activation='softmax')
])
```

⍰ Compile Model:

```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

⍰ Train Model:

```
model.fit(x_train, y_train, epochs=5)
```

⍰ Evaluate Model:

```
model.evaluate(x_test, y_test)
```

```
import joblib

# Save

joblib.dump(trained_model, 'startup_model.pkl')

# Load

model = joblib.load('startup_model.pkl')
```

Result:

Model Trained Successfully and saved successfully using joblib library.

Conclusion:

TensorFlow and Keras together form a powerful combination for building and deploying machine learning and deep learning models.
TensorFlow provides a scalable backend for computations, while Keras simplifies the process of defining and training models, making deep learning more accessible to everyone—from beginners to AI researchers.