**In-Class Exercise: Decision Tree Classification with the Mushroom Dataset**

**Objective**

In this exercise, you will implement a Decision Tree Classifier using the Mushroom Dataset. The goal is to predict whether a mushroom is edible or poisonous based on its features. you will gain hands-on experience with preprocessing data, building a model, and evaluating its performance.

**Exercise Instructions**

**Introduction**

A **decision tree** is a machine learning model used for classification and regression tasks. It splits the data into smaller subsets based on feature values, creating a tree-like structure. The "leaves" represent class labels or outcomes, while the "branches" represent decisions or conditions.

You will follow these steps:

1. **Load the Dataset**  
   Understand the dataset structure and what it represents.
2. **Preprocess the Data**  
   Prepare the data for modeling by converting categorical variables to numerical ones.
3. **Split the Data**  
   Divide the dataset into training and testing sets to evaluate model performance.
4. **Train a Decision Tree Model**  
   Fit a decision tree to the training data.
5. **Evaluate the Model**  
   Measure the model’s accuracy and interpret the results.
6. **Visualize the Tree**  
   Visualize the decision tree to understand how it works.

**Step-by-Step Guide with Definitions**

**Step 1: Import Libraries**

Python has powerful libraries for machine learning. Import the following:

* pandas: To load and manipulate data.
* sklearn (scikit-learn): To build and evaluate the decision tree.
* matplotlib: To visualize the tree.

**Task:** **Import Libraries**:

**Step 2: Load the Dataset**

The **Mushroom Dataset** contains information about mushrooms (e.g., cap shape, odor). The target variable, class, indicates whether a mushroom is edible (e) or poisonous (p).

**Task:** Load the dataset and explore it.

url = "https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data"

columns = ["class", "cap-shape", "cap-surface", "cap-color", "bruises", "odor",

"gill-attachment", "gill-spacing", "gill-size", "gill-color",

"stalk-shape", "stalk-root", "stalk-surface-above-ring",

"stalk-surface-below-ring", "stalk-color-above-ring",

"stalk-color-below-ring", "veil-type", "veil-color",

"ring-number", "ring-type", "spore-print-color", "population", "habitat"]

data = pd.read\_csv(url, header=None, names=columns)

**Step 3: Preprocess the Data**

Machine learning models cannot directly handle categorical data, so we convert it to numerical values using **label encoding**.

**Task:** Encode the data.

**Step 4: Split the Data**

To evaluate the model's performance, we divide the dataset into:

* **Training Set (80%)**: Used to train the model.
* **Testing Set (20%)**: Used to evaluate the model.

**Task:** Perform the split.

**Step 5: Train the Model**

A **Decision Tree Classifier** learns the relationship between features and the target variable.

* **Criterion**: Specifies how splits are made. Use "entropy" (based on information gain).
* **Max Depth**: Limits the depth of the tree to prevent overfitting.

**Task:** Train the model.

**Step 6: Evaluate the Model**

Model performance is evaluated using:

* **Accuracy**: Percentage of correct predictions.
* **Classification Report**: Precision, recall, and F1-score.
* **Confusion Matrix**: Breakdown of predictions by actual vs. predicted classes.

**Task:** Evaluate the model.

**Step 7: Visualize the Tree**

Visualizing the decision tree helps interpret how it makes predictions.

**Task:** Plot the tree.

**In-Class Exercise Questions**

1. What preprocessing steps were necessary, and why?
2. What does the max\_depth parameter do in the decision tree?
3. Interpret the confusion matrix: What do the values represent?
4. Experiment with different values of max\_depth and observe how it affects accuracy.
5. Try using "gini" as the criterion instead of "entropy". What changes in the results?