Lesson: Division of a Dataset

What is Dataset Division?

"In machine learning, we divide datasets into two parts:

1. **Training Data:** This is used to teach the computer how to find patterns and relationships in the data.
2. **Testing Data:** This is used to see if the computer has learned well.

Think of it as practice and a test:

- Training data is like practicing math problems.
- Testing data is like taking a math test to see if you understood."

Example 1: Predicting Student Scores

Dataset

Hours Studied	Score
1	20
2	40
3	60
4	80

Code

```
from sklearn.model_selection import train_test_split

# Step 1: Create the dataset
features = [[1], [2], [3], [4]]  # Features: Hours studied
labels = [20, 40, 60, 80]        # Labels: Scores

# Step 2: Split the dataset
X_train, X_test, y_train, y_test = train_test_split(features, labels,
test_size=0.25, random_state=42)

# Step 3: Print the results
print("Training Features:", X_train)
print("Testing Features:", X_test)
print("Training Labels:", y_train)
print("Testing Labels:", y_test)
```

Explanation

1. **`features` and `labels`:**
   - `features` contains the hours studied.
   - `labels` contains the scores.
2. **`test_size=0.25`:**
   - This means 25% of the data is for testing, and 75% is for training.

Example 2: Importing a CSV File (Car Prices)

Dataset: *Car Prices in a CSV File*

Model Year	Mileage (km)	Price ($)
2015	50,000	15,000
2017	30,000	20,000
2018	20,000	25,000
2020	10,000	30,000

CSV File Creation

Save the following data as a CSV file (e.g., `car_prices.csv`):

```
Model_Year,Mileage,Price
2015,50000,15000
2017,30000,20000
2018,20000,25000
2020,10000,30000
```

Code to Split the CSV File into Training and Testing Sets

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Step 1: Load the dataset from the CSV file
data = pd.read_csv('car_prices.csv')

# Step 2: Separate features and labels
features = data[['Model_Year', 'Mileage']]  # Features: Model Year and Mileage
labels = data['Price']                      # Labels: Price

# Step 3: Split the dataset
X_train, X_test, y_train, y_test = train_test_split(features, labels,
test_size=0.5, random_state=42)

# Step 4: Print the results
print("Training Features:")
print(X_train)
print("\nTesting Features:")
```

```
print(X_test)
print("\nTraining Labels:")
print(y_train)
print("\nTesting Labels:")
print(y_test)
```

Explanation

1. **`pd.read_csv('car_prices.csv')`:**
   - Reads the CSV file and loads it into a `DataFrame`.
2. **`features` and `labels`:**
   - `features` contains `Model_Year` and `Mileage`.
   - `labels` contains the `Price`.
3. **`test_size=0.5`:**
   - Splits the data into 50% training and 50% testing sets.
4. **Output:**
   - Training and testing sets are printed to show the split.

Example 3: Predicting House Prices

Dataset

Size (sq ft)	Rooms	Price ($)
1000	2	200,000
1500	3	300,000
2000	4	400,000
2500	4	500,000

Code

```
from sklearn.model_selection import train_test_split

# Step 1: Create the dataset
features = [[1000, 2], [1500, 3], [2000, 4], [2500, 4]]  # Features: Size and
Rooms
labels = [200000, 300000, 400000, 500000]                # Labels: Prices

# Step 2: Split the dataset
X_train, X_test, y_train, y_test = train_test_split(features, labels,
test_size=0.5, random_state=42)

# Step 3: Print the results
print("Training Features:", X_train)
print("Testing Features:", X_test)
print("Training Labels:", y_train)
print("Testing Labels:", y_test)
```

Example 4: Importing a Large CSV File

Dataset: *Housing Prices Data (CSV File)*

The dataset can be downloaded from online sources like Kaggle (https://www.kaggle.com/datasets/yasserh/housing-prices-dataset) . Save the file as `Housing.csv`.

Code to Split the Housing Dataset

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Step 1: Load the CSV data into a DataFrame
data = pd.read_csv('Housing.csv')

# Step 2: Display the first few rows of the dataset (optional)
print("Dataset Preview:")
print(data.head())

# Step 3: Split the dataset into features and labels
# Features are all columns except 'price', which is the target
features = data.drop(columns=['price'])
labels = data['price']

# Step 4: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.3, random_state=42)

# Step 5: Print the split data sizes
print("\nTraining Features Size:", X_train.shape)
print("Testing Features Size:", X_test.shape)
print("Training Labels Size:", y_train.shape)
print("Testing Labels Size:", y_test.shape)

# Step 6: Display sample training data (optional)
print("\nTraining Features Sample:")
print(X_train.head())
print("\nTraining Labels Sample:")
print(y_train.head())
```

Explanation:

1. **`pd.read_csv('Housing.csv')`:**
   Reads the CSV file containing the dataset into a Pandas DataFrame.

2. **`data.drop(columns=['price']):`**
   Excludes the `price` column from the features since it is the target label.
3. **`train_test_split:`**
   Splits the data into training (70%) and testing (30%) sets.
   - `random_state=42` ensures reproducibility.
   - `test_size=0.3` means 30% of the data is used for testing.
4. **`X_train, X_test, y_train, y_test:`**
   These variables store the training and testing features and labels.

Example : Penguins Dataset (Classification of Penguin Species)

The Penguins dataset contains data on penguin species along with their measurements like bill length and depth.

Python Code

```
import seaborn as sns

from sklearn.model_selection import train_test_split

import pandas as pd


# Step 1: Load the Penguins dataset from seaborn

penguins = sns.load_dataset('penguins')


# Step 2: Drop rows with missing values

penguins = penguins.dropna()


# Step 3: Convert categorical labels to numeric values

penguins['species'] = penguins['species'].astype('category').cat.codes

penguins['island'] = penguins['island'].astype('category').cat.codes

penguins['sex'] = penguins['sex'].astype('category').cat.codes


# Step 4: Display the first few rows of the dataset

print("Penguins Dataset Preview:")

print(penguins.head())


# Step 5: Split the dataset into features and labels

# Features: All columns except 'species'
```

```
features = penguins.drop(columns=['species'])
labels = penguins['species']

# Step 6: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.3, random_state=42)

# Step 7: Display the sizes of the split data
print("\nTraining Features Size:", X_train.shape)
print("Testing Features Size:", X_test.shape)
print("Training Labels Size:", y_train.shape)
print("Testing Labels Size:", y_test.shape)

# Step 8: Display sample training data
print("\nTraining Features Sample:")
print(X_train.head())
print("\nTraining Labels Sample:")
print(y_train.head())
```

Explanation

1. **Seaborn's `load_dataset`** provides the Penguins dataset.
2. Rows with missing data are removed using `dropna`.
3. The species column, which is categorical, is converted to numeric codes (e.g., Adelie -> 0, Chinstrap -> 1, Gentoo -> 2) for classification.
4. Data is split into 70% training and 30% testing.

Example 1: Iris Dataset (Built-in Dataset in Scikit-Learn)

The Iris dataset is one of the most famous datasets for classification. It contains information about iris flower measurements and their species.

Python Code

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
import pandas as pd
```

```
# Step 1: Load the Iris dataset
iris = load_iris()

# Step 2: Convert the dataset into a Pandas DataFrame
iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
iris_df['species'] = iris.target

# Step 3: Display the first few rows of the dataset
print("Iris Dataset Preview:")
print(iris_df.head())

# Step 4: Split the dataset into features and labels
# Features: All columns except 'species'
features = iris_df.drop(columns=['species'])
labels = iris_df['species']

# Step 5: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.25, random_state=42)

# Step 6: Display the sizes of the split data
print("\nTraining Features Size:", X_train.shape)
print("Testing Features Size:", X_test.shape)
print("Training Labels Size:", y_train.shape)
print("Testing Labels Size:", y_test.shape)

# Step 7: Display sample training data
print("\nTraining Features Sample:")
print(X_train.head())
print("\nTraining Labels Sample:")
print(y_train.head())
```

Explanation

1. The **`load_iris`** function loads the Iris dataset.
2. The dataset is converted to a Pandas DataFrame for easier manipulation.
3. Features are the measurements of the flowers (`sepal length`, `sepal width`, etc.), and the label (`species`) is the target variable.
4. The data is split into 75% training and 25% testing using `train_test_split`.