Practical 4: Linear Regression using Boston Housing Dataset - Cheatsheet

THEORY

What is Regression?

Regression is a supervised machine learning technique used for predicting a continuous value.

Linear regression is the simplest form where a straight line is fitted to the data.

Use Case:

Predicting house prices based on features such as number of rooms, age, location, etc.

Dataset:

Boston Housing Dataset (506 rows × 14 columns)

Target Variable: MEDV (Median value of owner-occupied homes)

Steps in Linear Regression:

1. Load the dataset.
2. Separate independent (X) and dependent (y) variables.
3. Split into training and testing datasets.
4. Train a Linear Regression model.
5. Predict and evaluate using RMSE and R^2 score.

CODE + EXPLANATION

1. Import Libraries

```
import pandas as pd       # For data manipulation
import numpy as np        # For numerical operations
from sklearn.model_selection import train_test_split  # For splitting dataset
from sklearn.linear_model import LinearRegression     # Linear Regression model
from sklearn.metrics import mean_squared_error        # Error metric
```

2. Load the Dataset

```
df = pd.read_csv('boston-housing-dataset.csv')
df                    # Display the dataset
```

3. View All Column Names

```
df.columns
```

4. Select Independent Variables (features)

```
x = df[['Unnamed: 0', 'CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS',
       'RAD', 'TAX', 'PTRATIO', 'B', 'LSTAT']]
```

5. Select Dependent Variable (target)

```
y = df['MEDV']
```

6. Split Dataset into Training and Testing Sets

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=42)
```

7. Create and Train the Model

```
model = LinearRegression()
model.fit(x_train, y_train)
```

8. Predict the Output

```
y_predict = model.predict(x_test)
y_predict
```

9. Evaluate Model Accuracy

```
model.score(x_train, y_train)   # R² score for training set
model.score(x_test, y_test)     # R² score for test set
```

10. Calculate RMSE (Root Mean Squared Error)

```
np.sqrt(mean_squared_error(y_test, y_predict))
```