**Exp 4**

User

**You**

**import pandas as pd**

**import numpy as np**

**from sklearn import linear\_model**

**from sklearn.model\_selection import train\_test\_split**

ChatGPT

**ChatGPT**

Sure, here's a brief explanation for each line:

1. **import pandas as pd**: Imports the Pandas library and aliases it as **pd**, allowing us to refer to it using **pd** throughout the code.
2. **import numpy as np**: Imports the NumPy library and aliases it as **np**, allowing us to refer to it using **np** throughout the code.
3. **from sklearn import linear\_model**: Imports the linear\_model module from the Scikit-learn library, which provides various linear models for regression and classification tasks.
4. **from sklearn.model\_selection import train\_test\_split**: Imports the **train\_test\_split** function from the **model\_selection** module in Scikit-learn. This function is used to split datasets into training and testing sets for model evaluation.

User

**You**

**from sklearn.datasets import load\_boston**

**boston = load\_boston()**

**print(boston)**

ChatGPT

**ChatGPT**

Here's a breakdown:

1. **from sklearn.datasets import load\_boston**: Imports the **load\_boston** function from the **datasets** module in Scikit-learn. This function is used to load the Boston Housing dataset, which is a commonly used dataset for regression tasks.
2. **boston = load\_boston()**: Calls the **load\_boston** function to load the dataset and assigns it to the variable **boston**.
3. **print(boston)**: Prints the contents of the **boston** dataset, which typically includes features, target values, and other information about the dataset.

User

**You**

**boston.feature\_names**

ChatGPT

**ChatGPT**

This line accesses the feature names of the Boston Housing dataset. In other words, it retrieves the names of the variables (features) that describe different aspects of the housing data, such as "CRIM" for per capita crime rate by town, "ZN" for proportion of residential land zoned for large lots, and so on. These feature names provide information about what each column in the dataset represents.

User

**You**

**boston.target**

ChatGPT

**ChatGPT**

This line retrieves the target variable from the Boston Housing dataset. In regression tasks like predicting house prices, the target variable is the one we want to predict. In the Boston Housing dataset, the target variable represents the median value of owner-occupied homes in $1000s. So, **boston.target** would give you an array containing the median house prices for the corresponding samples in the dataset.

User

**You**

**x = pd.DataFrame(boston.data, columns = boston.feature\_names)**

**y = pd.DataFrame(boston.target)**

ChatGPT

**ChatGPT**

Here's a breakdown:

1. **x = pd.DataFrame(boston.data, columns=boston.feature\_names)**: Creates a DataFrame **x** using Pandas, where the data is taken from **boston.data**, which contains the feature values of the Boston Housing dataset. The **columns** parameter assigns the feature names from **boston.feature\_names** to the columns of the DataFrame, ensuring each feature is appropriately labeled.
2. **y = pd.DataFrame(boston.target)**: Creates a DataFrame **y** using Pandas, where the data is taken from **boston.target**, which contains the target variable (median house prices). This DataFrame contains only one column, representing the target variable.

User

**You**

**x.head(10)**

ChatGPT

**ChatGPT**

The **head()** function in Pandas is used to display the first few rows of a DataFrame. **x.head(10)** will display the first 10 rows of the DataFrame **x**, showing the feature values of the Boston Housing dataset for the first 10 samples. This can give an initial glimpse into the structure and content of the dataset.

User

**You**

**y.head(10)**

ChatGPT

**ChatGPT**

The **y.head(10)** command will display the first 10 rows of the DataFrame **y**, which contains the target variable values (median house prices) for the corresponding samples in the Boston Housing dataset. This output will show the median house prices for the first 10 samples in the dataset.

User

**You**

**reg = linear\_model.LinearRegression()**

ChatGPT

**ChatGPT**

This line creates an instance of the **LinearRegression** class from the **linear\_model** module in Scikit-learn. This class represents a linear regression model, which is a type of predictive model used to establish the relationship between independent variables (features) and a dependent variable (target) by fitting a linear equation to observed data.

User

**You**

**x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.20, random\_state=42)**

ChatGPT

**ChatGPT**

This line splits the dataset into training and testing sets for both features (**x**) and the target variable (**y**). Here's what each variable represents:

* **x\_train**: The training set of features (input variables) used to train the model.
* **x\_test**: The testing set of features (input variables) used to evaluate the trained model's performance.
* **y\_train**: The corresponding training set of target variable values (output variable) used to train the model.
* **y\_test**: The corresponding testing set of target variable values (output variable) used to evaluate the trained model's performance.

The **train\_test\_split()** function from Scikit-learn is used for this purpose. It randomly splits the dataset into training and testing sets, where **test\_size=0.20** specifies that 20% of the data will be used for testing, and **random\_state=42** ensures reproducibility by fixing the random seed to 42.

User

**You**

**reg.fit(x\_train, y\_train)**

ChatGPT

**ChatGPT**

This line fits (trains) the linear regression model (**reg**) using the training data. The **fit()** method in Scikit-learn is used to train the model by finding the best parameters (coefficients) that minimize the difference between the actual target values (**y\_train**) and the predicted values obtained from the features (**x\_train**). Once this process is complete, the model is ready to make predictions on new data.

User

**You**

**print(reg.coef\_)**

ChatGPT

**ChatGPT**

This line prints the coefficients of the trained linear regression model (**reg**). The coefficients represent the weights assigned to each feature in the model. Each coefficient indicates the change in the target variable for a one-unit change in the corresponding feature, holding all other features constant. So, **reg.coef\_** will display an array containing the coefficients for each feature in the model.

User

**You**

**y\_pred = reg.predict(x\_test)**

**print(y\_pred)**

ChatGPT

**ChatGPT**

This line uses the trained linear regression model (**reg**) to make predictions on the testing set features (**x\_test**). The **predict()** method in Scikit-learn is applied to the linear regression model to generate predicted target variable values (**y\_pred**) based on the input features (**x\_test**). Finally, **print(y\_pred)** will display the predicted target variable values for the testing set samples.

Top of Form