Dr. A. Q. Khan Institute of Computer Sciences & Information Technology, (KICSIT)
Department of Computer Engineering

Artificial Intelligence Lab (04)

Introduction to Scikit-learn Library and Implementation of Linear Regression Using Buit-in Machine Learning Algorithms using Scikit-learn

Scikit-learn, often referred to as **sklearn**, is a popular Python library for machine learning. It is an open-source library that provides a wide range of tools and algorithms for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and more. Scikit-learn is built on top of other scientific libraries like NumPy, SciPy, and matplotlib and is designed to be easy to use and integrate with other data science libraries.

Here is an introduction to scikit-learn and its common use in machine learning:

Key Features and Use Cases:

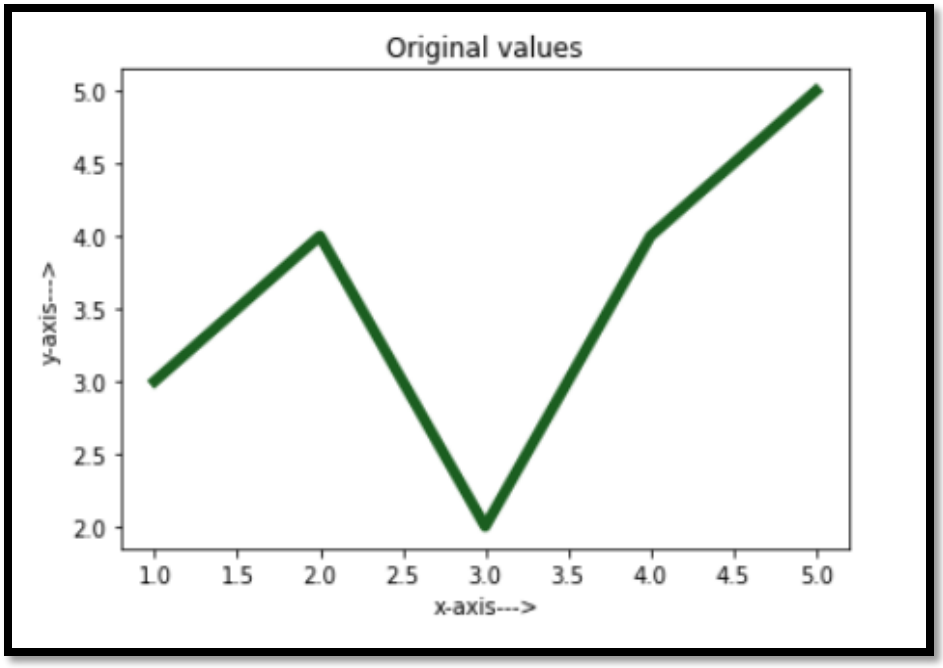
1. **User-Friendly API**: Scikit-learn provides a simple and consistent API that makes it easy to implement machine learning algorithms. This consistency is a major strength for newcomers to machine learning.
2. **Wide Range of Algorithms**: Scikit-learn offers a comprehensive selection of machine learning algorithms. Some of the most commonly used algorithms include linear and logistic regression, support vector machines, decision trees, random forests, k-means clustering, and more.
3. **Data Preprocessing**: Scikit-learn provides tools for data preprocessing, including data scaling, feature selection, and handling missing values. This is crucial in preparing data for machine learning tasks.
4. **Model Selection and Evaluation**: The library offers functions for model selection through techniques like cross-validation and grid search. It also provides tools for evaluating model performance with metrics like accuracy, precision, recall, and F1-score.
5. **Dimensionality Reduction**: Scikit-learn includes dimensionality reduction techniques such as Principal Component Analysis (PCA) and feature extraction methods.
6. **Ensemble Methods**: Scikit-learn supports ensemble methods, including bagging and boosting, which can improve model performance by combining the predictions of multiple base models.
7. **Integration with Other Libraries**: Scikit-learn can easily integrate with other popular data science libraries like pandas, NumPy, and matplotlib. This makes it a versatile tool for building end-to-end machine learning pipelines.

Lab Instructor: Engr. Zia Ur Rehman

Example 01

We have some values of x or input or independent variable given and on behalf of these values we also have values of y or output or dependent variable, and now we want to predict values of y on new values of x.

x	*y (Actual Values)*
1	3
2	4
3	2
4	4
5	5
6	?
7	?
8	?



Implementing Linear Regression Using Built-in Model

01. Importing Libraries

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score
```

02. Creating arrays for x (Hours Studied) and y (Final Exam Score)

In [3]:
```
x = np.array([2,3,4,5,6,7])
y = np.array([60,70,80,85,90,95])

print(f"Hours Studied: {x}\nFinal Exam Score: {y}")
```

```
Hours Studied: [2 3 4 5 6 7]
Final Exam Score: [60 70 80 85 90 95]
```

convert x and y to column vector

In [3]:
```
x = np.array([2,3,4,5,6,7]).reshape(-1,1)
y = np.array([60,70,80,85,90,95]).reshape(-1,1)

print(f"Hours Studied: {x}\nFinal Exam Score: {y}")
```

```
Hours Studied: [[2]
 [3]
 [4]
 [5]
 [6]
 [7]]
Final Exam Score: [[60]
 [70]
 [80]
 [85]
 [90]
 [95]]
```

03. Select the Model

In [14]:
```
linearModel = LinearRegression()
```

04. Train the Model

In [15]:
```
linearModel.fit(x,y)
```

LinearRegression()

05. Test the Model

In [16]:
```
test = np.array([3.5, 5, 7, 8, 9, 10]).reshape(-1,1)
test
```

Out[16]:
```
array([[ 3.5],
       [ 5. ],
       [ 7. ],
       [ 8. ],
       [ 9. ],
       [10. ]])
```

In [17]:
```
yPred = linearModel.predict(test)
yPred
```

Out[17]:
```
array([ 73.14285714,  83.42857143,  97.14285714, 104.        ,
       110.85714286, 117.71428571])
```

06. Find the Accuracy

In [18]:
```
linearModel.score(x,y)
```

Out[18]:
```
0.9680672268907564
```

```
from sklearn.metrics import mean_squared_error, accuracy_score
mSE = mean_squared_error(test, yPred)
rms = np.sqrt(mSE)
rms
```

Lab Task/Lab Report

Assigned Date:

1. **Lab Task 01**: Try to implement the Example 01 using built-in linear regression model of scikit-learn library in Python Programming Language.

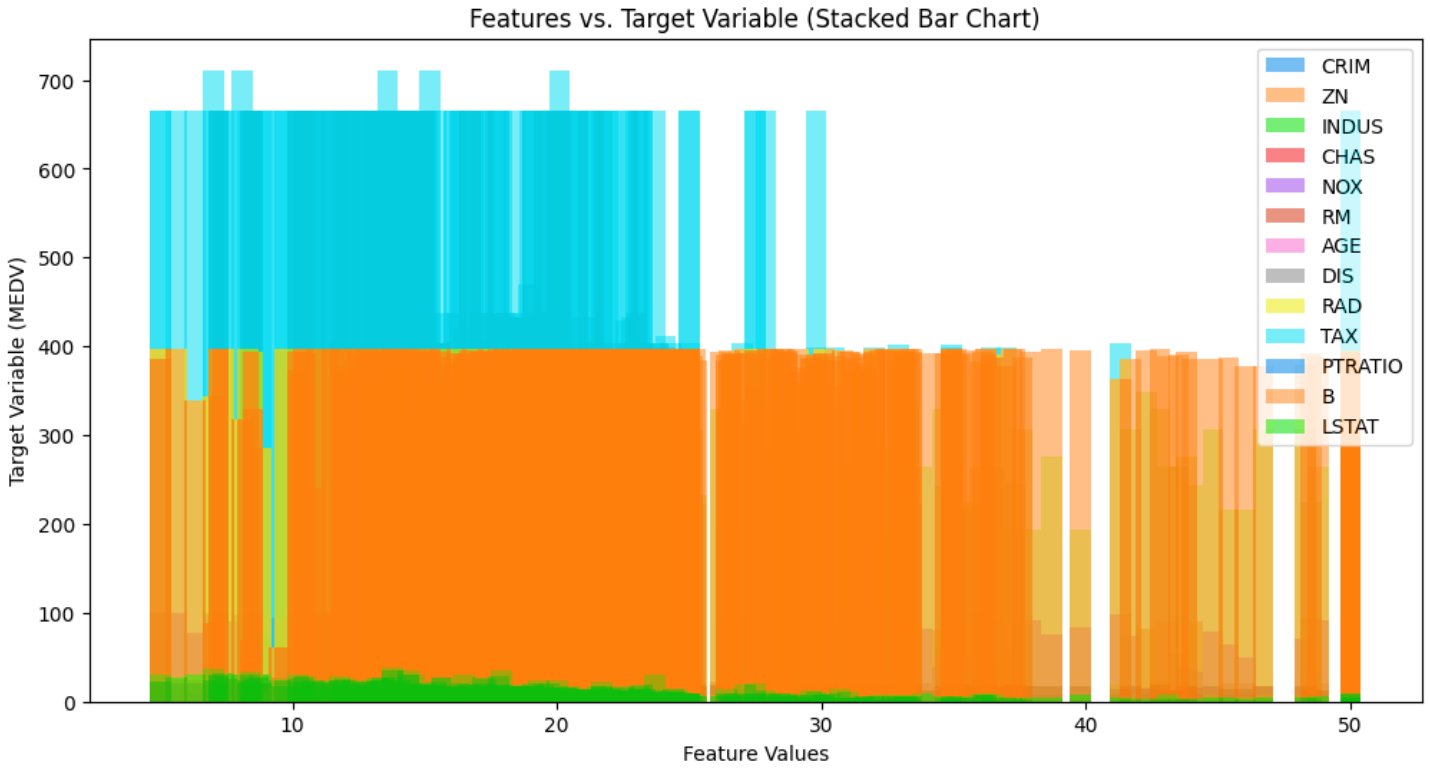
[CLO-01, PLO-02, P-3(Guided Response), Rubric (Coding)]

Marks	1	2	3	4
Coding	The code is not as per guidelines and requirements are not met	Some section of code is correct	Most section of code is correct and understands it well	The code is properly written, and have good understanding about it

2. **Lab Task 02**: The Boston dataset, also known as the **Boston Housing dataset**, is a well-known dataset that is included in scikit-learn (sklearn). It is often used for regression analysis and is based on data collected by the U.S. Census Service concerning housing in the area of Boston, Massachusetts. This dataset is often used for teaching and practicing regression models, especially linear regression. Here's a description of the Boston dataset:

Dataset Characteristics:

Data Type: The Boston dataset is a collection of structured data, where each record represents a different neighborhood in Boston.
Data Size: It contains 506 instances (neighborhoods) and 13 different features (attributes) or input variables.
Target Variable: The target variable in this dataset is the median value of owner-occupied homes (in thousands of dollars), which is typically used as the target for regression tasks.



Features (Attributes):

1. CRIM: Per capita crime rate by town.
2. ZN: Proportion of residential land zoned for large lots.
3. INDUS: Proportion of non-retail business acres per town.
4. CHAS: Charles River dummy variable (1 if tract bounds the river, 0 otherwise).
5. NOX: Nitrogen oxide concentration (parts per 10 million).
6. RM: Average number of rooms per dwelling.
7. AGE: Proportion of owner-occupied units built before 1940.
8. DIS: Weighted distance to employment centers.
9. RAD: Index of accessibility to radial highways.
10. TAX: Property tax rate (in thousands of dollars).
11. PTRATIO: Pupil-teacher ratio by town.
12. B: Proportion of residents of African American descent.
13. LSTAT: Percentage of lower-status population.

Target Variable:

- MEDV: Median value of owner-occupied homes in thousands of dollars.

Use Cases: The Boston dataset is primarily used for regression analysis and predictive modeling. Researchers and data scientists often use it to build and evaluate regression models that aim to predict the median value of homes based on the given features. It's used for educational purposes, benchmarking regression algorithms, and learning how to work with real-world datasets.

Make machine learning model for this scenario and implement in python using built-in model from Scikit-learn. If you need help in coding →click here

[CLO-02, PLO-03, P-4(Mechanism), Rubric (Model Implementation)]

Marks	1	2	3	4
Model Implementation	The model is not implemented as per guidelines and requirements are not met	Some section of model is correctly implemented	Most section of model is correctly implemented and understands it well	The model is properly implemented, and have good understanding about it

3. **Lab Task 03**: The Diabetes dataset in scikit-learn is known as the "Diabetes dataset" or "diabetes" and is often used for regression analysis and predictive modeling. This dataset contains ten baseline variables (age, sex, BMI, average blood pressure, and six blood serum measurements) and a quantitative measure of disease progression one year after baseline. Here is a description of the Diabetes dataset:

   Dataset Characteristics:

   Data Type: The Diabetes dataset is a collection of structured data. It is commonly used for regression analysis, where you aim to predict a continuous target variable based on one or more input features.
   Data Size: It contains 442 instances (samples) and 11 different features (attributes).

Features (Attributes):

1. Age: Age in years
2. Sex: A binary variable indicating gender (0 for female, 1 for male)
3. BMI: Body mass index
4. BP: Average blood pressure
5. S1: Total serum cholesterol
6. S2: Low-density lipoproteins (LDL cholesterol)
7. S3: High-density lipoproteins (HDL cholesterol)
8. S4: Log of serum triglycerides level
9. S5: Blood sugar level
10. S6: T-cell count (a measure of the immune system's response)

Target Variable:

- **Diabetes progression:** A quantitative measure of disease progression one year after baseline. This is a continuous variable, and the goal is to predict or model its value.

Use Cases:

The Diabetes dataset is typically used for regression analysis, especially in machine learning and statistical modeling courses. It's an example of a real-world dataset where the goal is to predict a continuous target variable (diabetes progression) based on the given features. Common use cases include:

- **Regression Modeling:** Building and evaluating regression models, such as linear regression, support vector regression, or decision tree regression, to predict diabetes progression.
- **Feature Selection:** Exploring feature importance and selecting relevant features for the regression model.
- **Model Evaluation:** Practicing model evaluation techniques, including mean squared error (MSE), R-squared, and cross-validation.

Make machine learning model for this scenario and implement in python using built-in model from Scikit-learn.

If you need help in coding →click here

[CLO-02, PLO-03, P-4(Mechanism), Rubric (Model Implementation)]

Marks	1	2	3	4
Model Implementation	The model is not implemented as per guidelines and requirements are not met	Some section of model is correctly implemented	Most section of model is correctly implemented and understands it well	The model is properly implemented, and have good understanding about it

Artificial Intelligence Lab
All Rubrics of Microprocessor & Interfacing Lab
CLO 1

Marks	1	2	3	4
Coding	The code is not as per guidelines and requirements are not met	Some section of code is correct	Most section of code is correct and understands it well	The code is properly written, and have good understanding about it

CLO 2

Marks	1	2	3	4
Model Implementation	The model is not implemented as per guidelines and requirements are not met	Some section of model is correctly implemented	Most section of model is correctly implemented and understands it well	The model is properly implemented, and have good understanding about it

CLO 3

Marks	1	2	3	4

Data Pre-processing	The data is not pre-processed as per guidelines and requirements are not met	Some section of data pre-processing is correct	Most section of data pre-processing is correct and understands it well	The data pre-processing is done properly, and have good understanding about it

CLO 4

Marks	**1**	**2**	**3**	**4**
Team Work	Rarely listens to, shares with, and supports the efforts of others. Often is not a good team member.	Often listen to, shares with and supports the efforts of others, but sometimes is not good team member.	Usually listen to, shares with, and supports the efforts of others. Usually, respectful and listening actively	Almost always listens to, shares with and supports the efforts of others. Tries to keep people working well together.

Lab Report Rubric: *must be submitted in next lab***.**

Marks	**1**	**2**	**3**	**4**
Lab Report	The lab report does not follow the guidelines for formatting.	Presents some sections of the lab in the correct order. Three or more sections are not in the correct order; missing heading or title;	Presents most sections of the lab in the correct order, one or two sections may not be in the correct order; heading or title missing or not complete;	Presents all the sections of the lab in the correct order with correct formatting: includes correct heading, section headings and title of lab;