Implement linear regression using Scikit-learn on a sample dataset

Table of Contents

Description

This lab guide will help you implement linear regression using Scikit-learn on the California Housing dataset. Linear regression is a basic and commonly used type of predictive analysis. The overall idea of regression is to examine two things:

- Does a set of predictor variables do a good job in predicting an outcome (dependent) variable?
- Which variables in particular are significant predictors of the outcome variable, and in what way do they impact the outcome variable?

Problem Statement

You will use the California Housing dataset to predict housing prices based on various features such as the median income of residents in the area, the average number of rooms, the median house age, and more.

Prerequisites

Completion of all previous lab guides (up to Lab Guide-01) is required before proceeding with Lab Guide-02.

Software Required

- Python version 3.11.9
- Visual Studio Code (VSCode)
- Libraries: `numpy`, `pandas`, `matplotlib`, `seaborn`, `scikit-learn`

Hardware Requirements

- Minimum 4GB RAM.
- At least 1GB of free disk space.
- A GPU (optional, but recommended for faster training).

Setup Instructions

Setting Up a Python Environment

1. Install Python

You can download and install Python 3.11.9 from the official Python website:

- Visit the **official Python website**.
- Locate a reliable version of Python 3, **"Download Python 3.11.9"**.
- Choose the correct link for your device from the options provided: either Windows installer (64-bit) or Windows installer (32-bit) and proceed to download the executable file.

2. Install Visual Studio Code (VSCode)

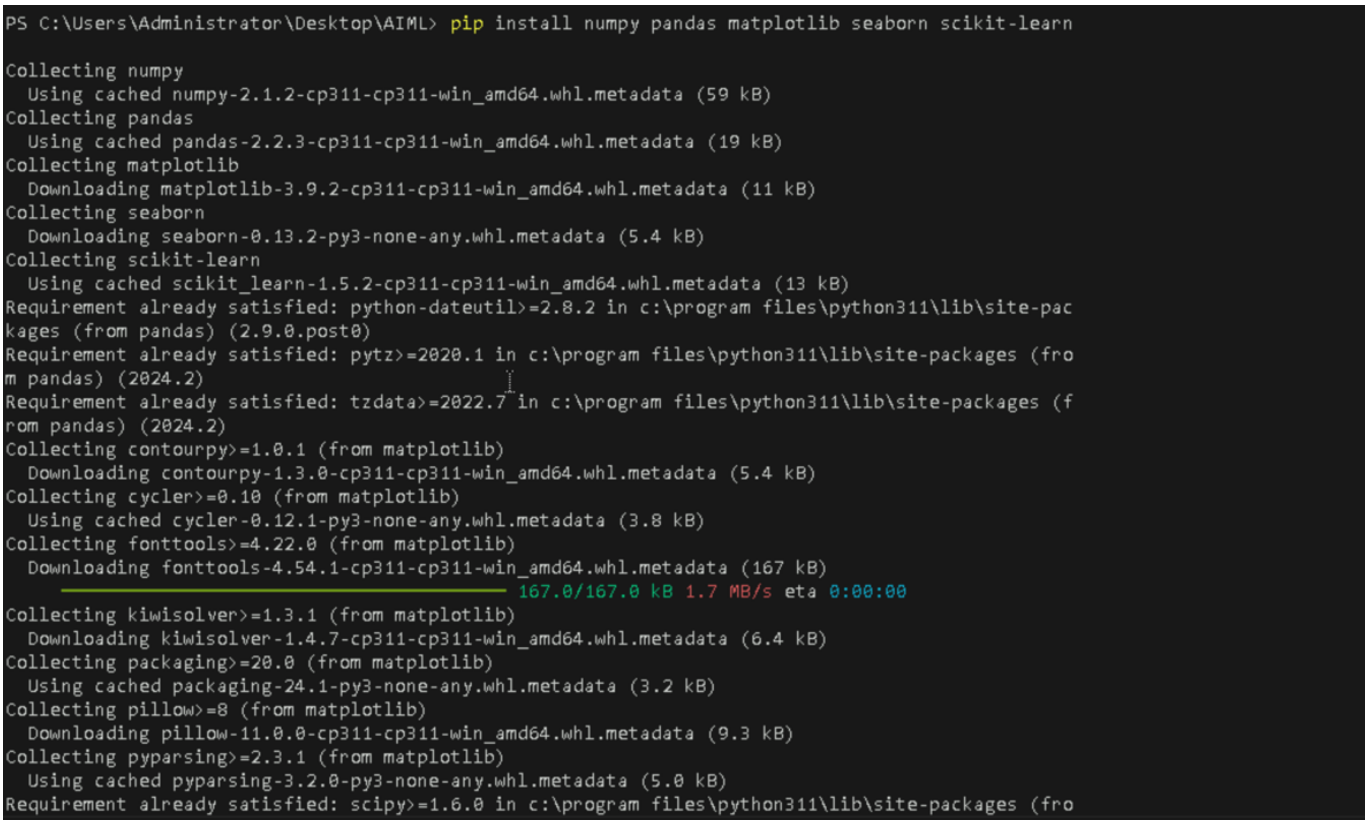
Download and install VSCode from the official Visual Studio Code website:

Download Visual Studio Code

3. Install Required Libraries

Open a terminal or command prompt and run the following commands to install the necessary libraries:

```
pip install numpy pandas matplotlib seaborn scikit-learn
```



Key Concepts

Linear Regression

Linear regression is a statistical method used to model the relationship between a dependent variable (target) and one or more independent variables (features). It attempts to find the linear relationship by fitting a line through the data points.

Coefficients

In linear regression, coefficients represent the relationship between each feature and the target variable. Each coefficient indicates how much the target variable is expected to change when the corresponding feature increases by one unit, holding all other features constant. A positive coefficient means that as the feature increases, the target variable tends to increase, while a negative coefficient indicates that as the feature increases, the target variable tends to decrease.

Mean Squared Error (MSE)

Mean Squared Error is a metric used to measure the average of the squares of the errors—that is, the average squared difference between predicted and actual values. Lower MSE values indicate better model performance.

R-squared (R^2)

R-squared is a statistical measure that represents the proportion of the variance for the target variable that's explained by the independent variables in the model. An R^2 value closer to 1 indicates a better fit for the model.

Matplotlib

Matplotlib is a Python library used for creating static, interactive, and animated visualizations in Python. It provides a wide variety of plotting functions to visualize data in various formats, making it easy to analyze and interpret results.

Example Usage

Loading the Dataset

- **Create a new python file**
  - Create a Python file named `lineary_regression.py`.

We will use the California Housing dataset from Scikit-learn. This dataset can be loaded as follows:

```
from sklearn.datasets import fetch_california_housing
import pandas as pd

# Load the dataset
housing = fetch_california_housing()
data = pd.DataFrame(housing.data, columns=housing.feature_names)
data['target'] = housing.target

# Display the first few rows of the dataset
print(data.head())
```

Run the Python file

- Use the command below in your terminal to run the Python file:

```
python linear_regression.py
```

Output

```
PS C:\Users\Administrator\Desktop\AIML> python linear_regression.py
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  Longitude  target
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88    -122.23   4.526
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86    -122.22   3.585
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85    -122.24   3.521
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85    -122.25   3.413
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85    -122.25   3.422
```

Implementing Linear Regression

We will use Scikit-learn to implement linear regression on this dataset.

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Load the dataset
housing = fetch_california_housing()
data = pd.DataFrame(housing.data, columns=housing.feature_names)
data['target'] = housing.target


# Split the data into training and testing sets
X = data.drop('target', axis=1)
y = data['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# Create and train the model
model = LinearRegression()
model.fit(X_train, y_train)
# Make predictions
y_pred = model.predict(X_test)
# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R^2 Score: {r2}")
```

Run the Python file

- Use the command below in your terminal to run the Python file:

```
python linear_regression.py
```

Output

```
PS C:\Users\Administrator\Desktop\AIML> python linear_regression.py
Mean Squared Error: 0.555891598695244
R^2 Score: 0.5757877060324511
```

True vs. Predicted Values

We can visualize the relationship between the true and predicted values and also look at the distribution of the target variable.

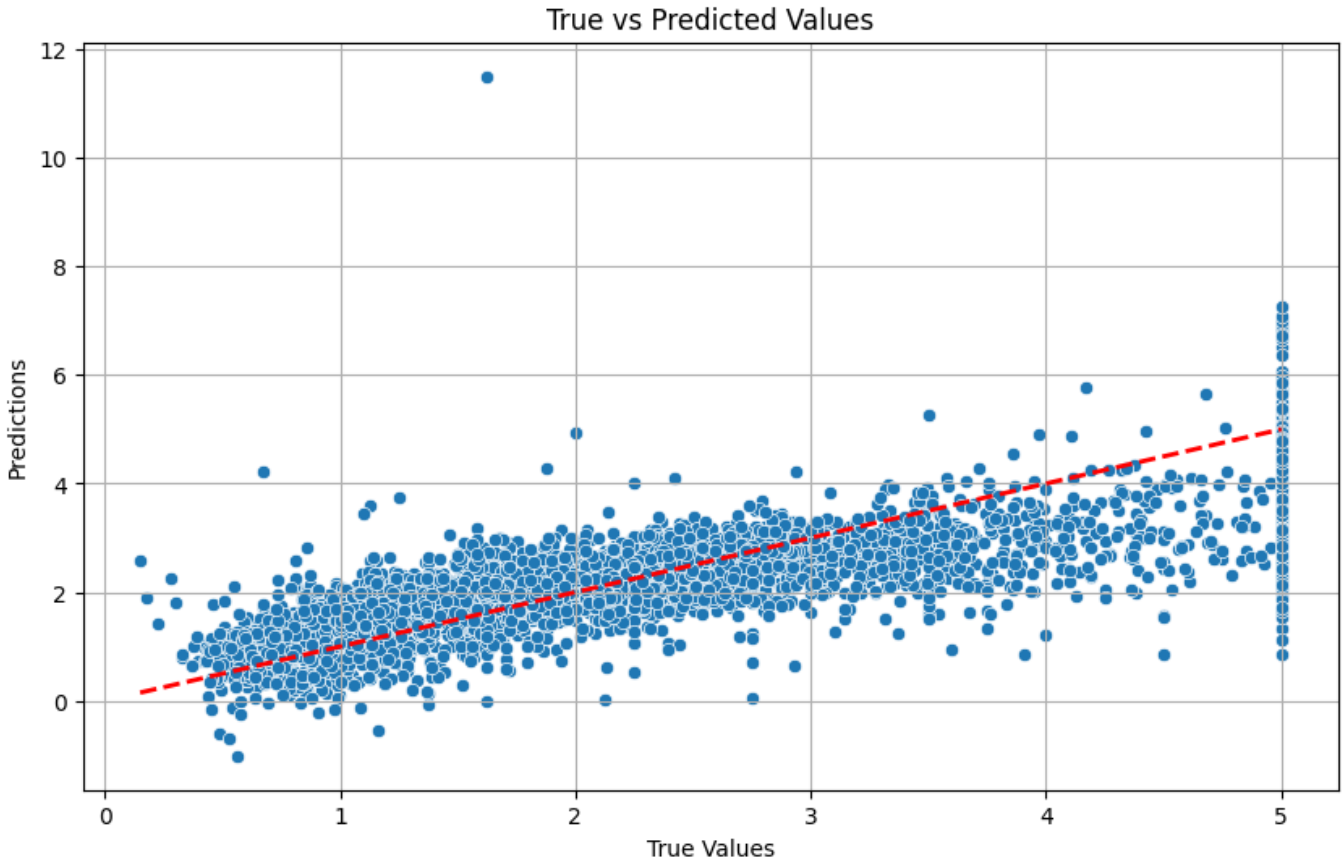
```
import matplotlib.pyplot as plt
import seaborn as sns

# Plot true vs predicted values
plt.figure(figsize=(10, 6))
sns.scatterplot(x=y_test, y=y_pred)
plt.plot([y.min(), y.max()], [y.min(), y.max()], 'r--', lw=2)  # Diagonal line
plt.xlabel('True Values')
plt.ylabel('Predictions')
plt.title('True vs Predicted Values')
plt.grid()
plt.show()
```

Run the Python file

- Use the command below in your terminal to run the Python file:

```
python linear_regression.py
```

Output



Distribution of Target Variable

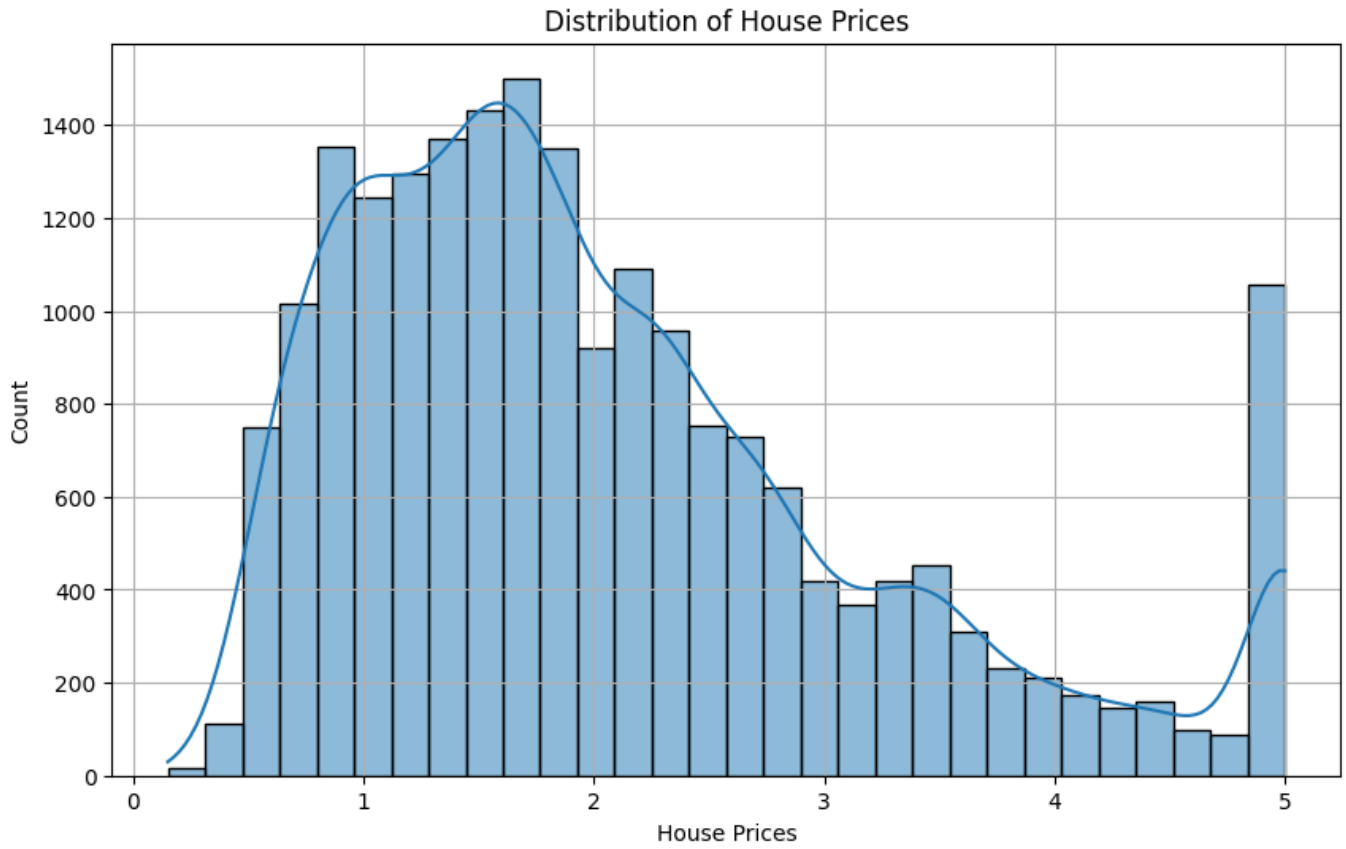
```
# Plot distribution of target variable
plt.figure(figsize=(10, 6))
sns.histplot(data['target'], bins=30, kde=True)
plt.xlabel('House Prices')
plt.title('Distribution of House Prices')
plt.grid()
plt.show()
```

Run the Python file

- Use the command below in your terminal to run the Python file:

```
python linear_regression.py
```

Output



Residual Plot

A residual plot helps visualize the errors of the model.

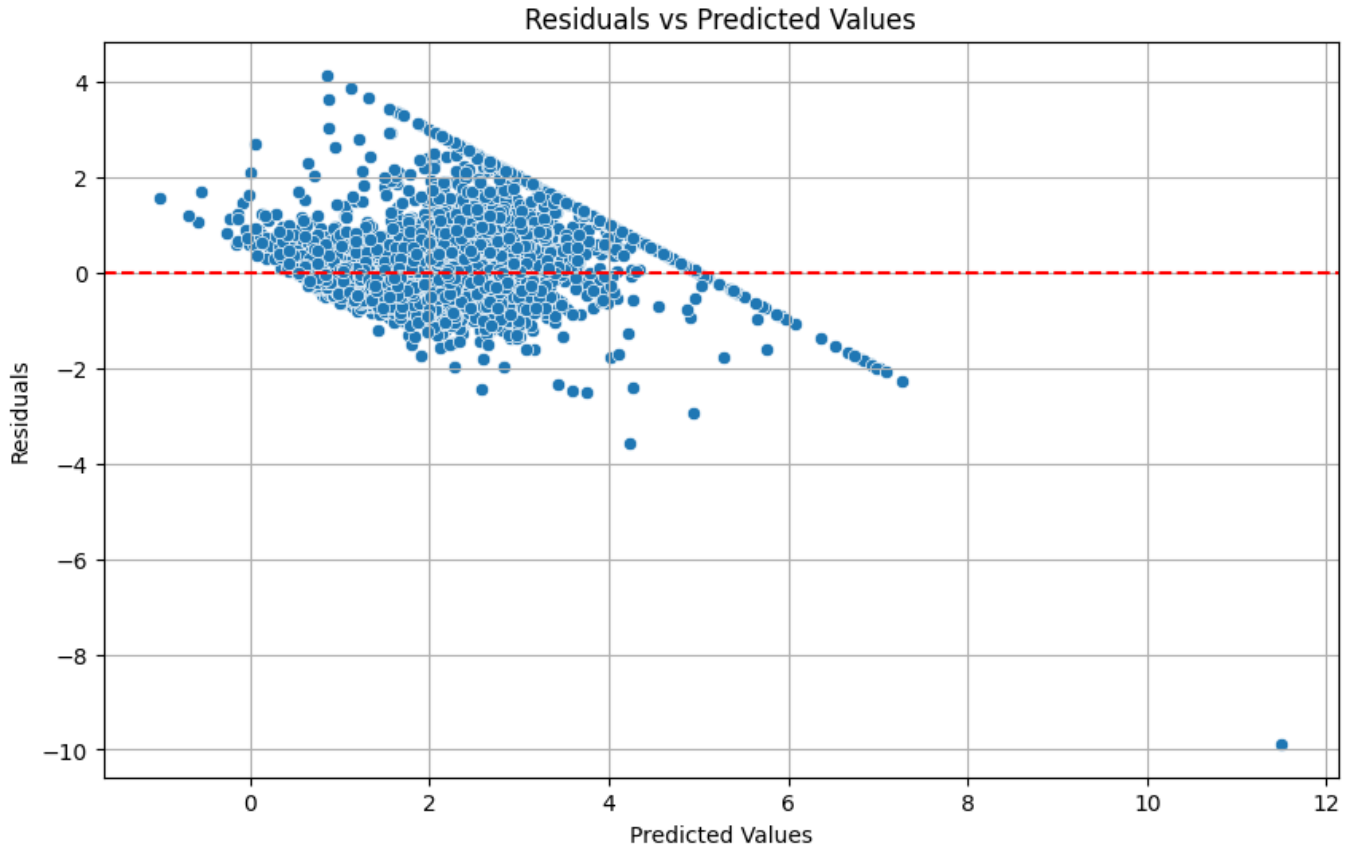
```
# Plot residuals
residuals = y_test - y_pred
plt.figure(figsize=(10, 6))
sns.scatterplot(x=y_pred, y=residuals)
plt.axhline(0, color='red', linestyle='--')
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.title('Residuals vs Predicted Values')
plt.grid()
plt.show()
```

Run the Python file

- Use the command below in your terminal to run the Python file:

```
python linear_regression.py
```

Output



Feature Importance

We can visualize the coefficients of the features to understand their importance.

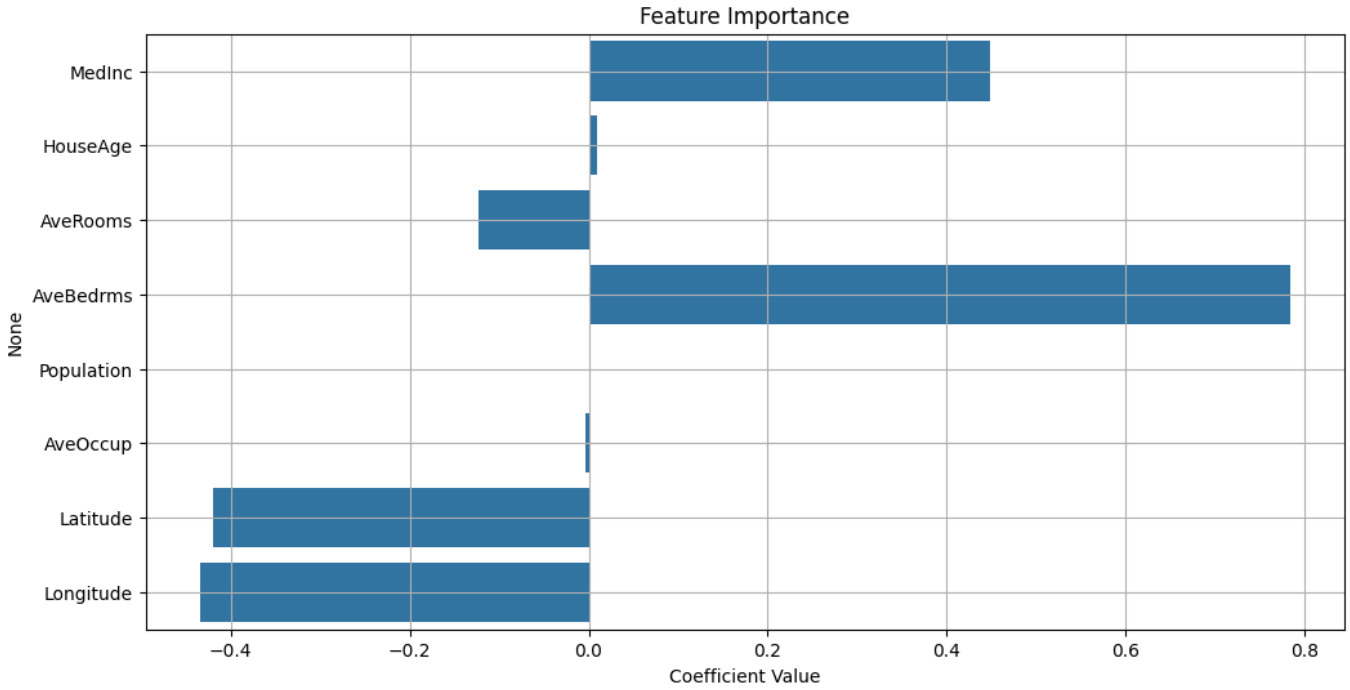
```
# Plot feature importance
plt.figure(figsize=(12, 6))
features = X.columns
importance = model.coef_
sns.barplot(x=importance, y=features)
plt.xlabel('Coefficient Value')
plt.title('Feature Importance')
plt.grid()
plt.show()
```

Run the Python file

- Use the command below in your terminal to run the Python file:

```
python linear_regression.py
```

Output



References

- Scikit-learn Documentation
- California Housing Dataset