**U19EC046 | ML | LAB 6**

**AIM**

Write a program to implement the Linear Regression for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets.

**ALGORITHM**

1. Import necessary libraries
2. Read the dataset using pandas
3. Split the dataset into features and results
4. Split the features and results into training and testing dataset using scikit learn

test\_train\_split

1. Apply pre-processing if needed, use scikit-learn preprocessing class
2. If your dataset has categorical features encode it using scikit learn’s one hot encoder
3. Create an instance of regressor
4. Train the model using fit method
5. Predict the result using predict method on model
6. Find the r2 score and mean square error using scikit-learn matrices

**CODE**

1. Linear Regression

|  |
| --- |
| **import numpy as np**  **import matplotlib.pyplot as plt**  **import pandas as pd**  **dataset = pd.read\_csv('Salary\_Data.csv')**  **X = dataset.iloc[:, :-1].values**  **y = dataset.iloc[:, -1].values**  **from sklearn.model\_selection import train\_test\_split**  **X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 543)**  **from sklearn.linear\_model import LinearRegression**  **regressor = LinearRegression()**  **regressor.fit(X\_train, y\_train)**  **y\_pred = regressor.predict(X\_test)**  **from sklearn.metrics import r2\_score, mean\_squared\_error**  **score = r2\_score(y\_test, y\_pred)**  **print(f"score: {round(score\*100, 3)} %")**  **mse = mean\_squared\_error(y\_true=y\_test, y\_pred=y\_pred)**  **print(f"mean squared error: {round(mse, 3)}")** |

1. Multiple Linear Regression

|  |
| --- |
| **import numpy as np**  **import matplotlib.pyplot as plt**  **import pandas as pd**  **dataset = pd.read\_csv('50\_Startups.csv')**  **X = dataset.iloc[:, :-1].values**  **y = dataset.iloc[:, -1].values**  **from sklearn.compose import ColumnTransformer**  **from sklearn.preprocessing import OneHotEncoder**  **ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [3])], remainder='passthrough')**  **X = np.array(ct.fit\_transform(X))**  **from sklearn.model\_selection import train\_test\_split**  **X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 755)**  **from sklearn.linear\_model import LinearRegression**  **regressor = LinearRegression()**  **regressor.fit(X\_train, y\_train)**  **y\_pred = regressor.predict(X\_test)**  **from sklearn.metrics import r2\_score, mean\_squared\_error**  **score = r2\_score(y\_test, y\_pred)**  **print(f"score: {round(score\*100, 3)} %")**  **mse = mean\_squared\_error(y\_true=y\_test, y\_pred=y\_pred)**  **print(f"mean squared error: {round(mse, 3)}")** |

1. Polynomial Regression

|  |
| --- |
| **import numpy as np**  **import matplotlib.pyplot as plt**  **import pandas as pd**  **dataset = pd.read\_csv('Position\_Salaries.csv')**  **X = dataset.iloc[:, 1:-1].values**  **y = dataset.iloc[:, -1].values**  **from sklearn.model\_selection import train\_test\_split**  **X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 755)**  **from sklearn.linear\_model import LinearRegression**  **lin\_reg = LinearRegression()**  **lin\_reg.fit(X\_train, y\_train)**  **from sklearn.preprocessing import PolynomialFeatures**  **from sklearn.metrics import r2\_score, mean\_squared\_error**  ***# Polynomial Regression***  **poly\_reg = PolynomialFeatures(degree = 13)**  **X\_poly = poly\_reg.fit\_transform(X\_train)**  **lin\_reg\_2 = LinearRegression()**  **lin\_reg\_2.fit(X\_poly, y\_train)**  **y\_pred\_poly = lin\_reg\_2.predict(poly\_reg.fit\_transform(X\_test))**  **score\_poly = r2\_score(y\_test, y\_pred\_poly)**  **print(f"r2 score Poly: {round(score\_poly\*100, 3)} %")**  **mse\_poly = mean\_squared\_error(y\_true=y\_test, y\_pred=y\_pred\_poly)**  **print(f"mean squared error Poly: {round(mse\_poly, 3)}")**  ***# Linear Regression***  **y\_pred\_lin = lin\_reg.predict(X)**  **score\_lin = r2\_score(y, y\_pred\_lin)**  **print(f"r2 score Lin: {round(score\_lin\*100, 3)} %")**  **mse\_lin = mean\_squared\_error(y\_true=y, y\_pred=y\_pred\_lin)**  **print(f"mean squared error Lin: {round(mse\_lin, 3)}")** |

**OUTPUT**

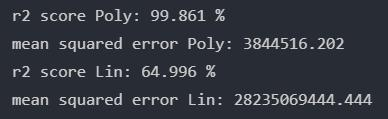
1. Linear Regression



1. Multiple Linear Regression



1. Polynomial Regression and comparing with linear regression



**CONCLUSION**

In This Practical, we have studied and implemented linear, multiple linear and polynomial regression using scikit learn library in python. It was observed the polynomial regression outperformed linear regression in non linear dataset.