Program on Linear Regression

Name	Pratik Daga
Roll No.	11
Experiment No.	01
Learning Objective:	Implement the linear regression technique with appropriate data set and application
Learning Outcome	Student are able to successfully implement linear regression algorithm
Course Outcome:	CSL701.4
Program Outcome:	(PO 3) Design/ development of solutions: Breadth and uniqueness of engineering problems i.e. the extent to which problems are original and to which solutions have previously been identified or codified (PO 12) Life Long Learning
Bloom's Taxonomy Level:	Analysis,Create

Theory:

Linear Regression -

Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.

This form of analysis estimates the coefficients of the linear equation, involving one or more independent variables that best predict the value of the dependent variable. Linear regression fits a straight line or surface that minimizes the discrepancies between predicted and actual output values. There are simple linear regression calculators that use a “least squares” method to discover the best-fit line for a set of paired data. You then estimate the value of X (dependent variable) from Y (independent variable).

Algorithm :

Algorithm for linear regression

y= mx + c

Here,

Y = Dependent Variable (Target Variable)

X = Independent Variable (predictor

Variable) m = Slope of the line

c = intercept of the line (Gives an additional degree of freedom)

Program

Cell 1 :

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
```

Cell 2 :

```
# observations / data
x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])
```

Cell 3 :

```
def estimate_coef(x, y):
    # number of observations/points
    n = np.size(x)
```

```
	# mean of x and y vector
	m_x = np.mean(x)
	m_y = np.mean(y)

	# calculating cross-deviation and deviation about x
	SS_xy = np.sum(y*x) - n*m_y*m_x
	SS_xx = np.sum(x*x) - n*m_x*m_x

	# calculating regression coefficients
	b_1 = SS_xy / SS_xx
	b_0 = m_y - b_1*m_x

	return (b_0, b_1)
```

Cell 4 :

```
# estimating coefficients
b = estimate_coef(x, y)
print("Estimated coefficients:\nb_0 = {} \
\nb_1 = {}".format(round(b[0],2), round(b[1],2)))
```

Cell 5 :

```
def plot_regression_line(x, y, b):
	# plotting the actual points as scatter plot
	plt.scatter(x, y, color = "m",
				marker = "o", s = 30)
```

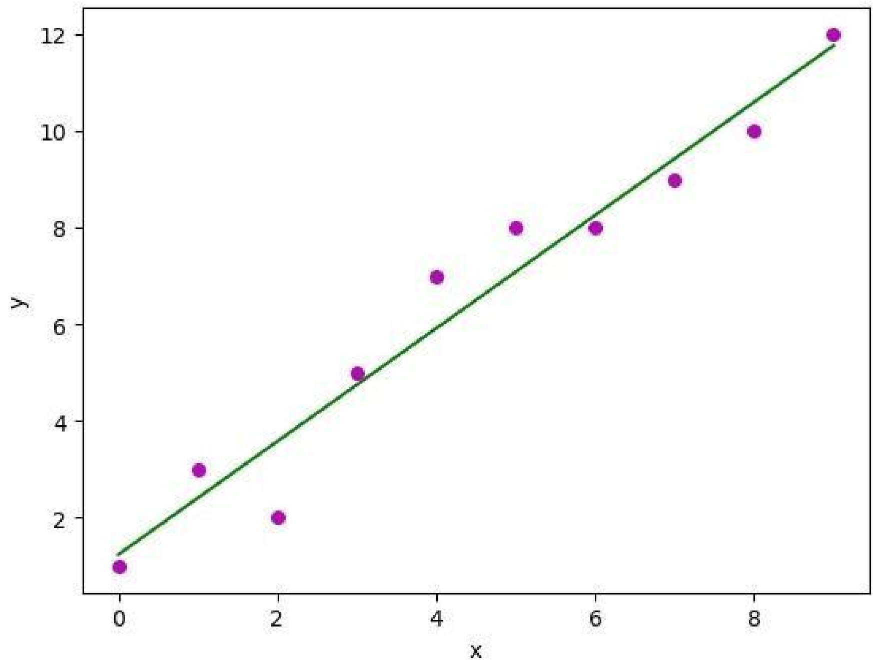
```
# predicted response vector
y_pred = b[0] + b[1]*x

# plotting the regression line
plt.plot(x, y_pred, color =
"g")

# putting labels
plt.xlabel('x')
plt.ylabel('y')

# function to show plot
plt.show()
```

Cell 6 :

```
# plotting regression line
plot_regression_line(x, y, b)
```

Outcome :

```
Estimated coefficients:
b_0 = 1.24
b_1 = 1.17
```



Data Set and Application – Non linear regression

Data Set:

1. Image Data: Linear regression is not suitable for image data because it cannot handle the complex spatial relationships present in images. Each pixel in an image is typically treated as an independent feature, disregarding the important spatial information.
2. Text Data: Linear regression is not ideal for text data as it fails to capture the inherent sequential and contextual information in the text. Natural Language Processing tasks like sentiment analysis, machine translation, and text generation require more sophisticated models like recurrent neural networks (RNNs) or transformers.
3. Time Series Data: Linear regression is not appropriate for time series data, which exhibits temporal dependencies and often has seasonality and trends. Specialized models such as ARIMA (AutoRegressive Integrated Moving Average) or seasonal decomposition methods are commonly used for time series forecasting.

4. Graph Data: Linear regression is not suitable for graph-structured data, where entities are represented as nodes and relationships as edges. Graph-based tasks like link prediction, community detection, or graph classification require graph neural networks (GNNs) or other graph-specific models.

Applications:

1. Object Detection: Linear regression cannot be used effectively for object detection tasks, where the goal is to locate and classify objects within an image. Object detection requires complex models like Faster R-CNN, YOLO (You Only Look Once), or SSD (Single Shot Multibox Detector) that use convolutional neural networks (CNNs) and bounding box regression.
2. Speech Recognition: Linear regression is not suitable for speech recognition tasks, which involve converting audio signals into text. Modern speech recognition systems, like those based on deep learning, utilize recurrent neural networks (RNNs) and attention mechanisms to capture the temporal dependencies in audio data.
3. Recommendation Systems: Linear regression is not the best choice for recommendation systems, where the goal is to predict user preferences based on historical behavior. Collaborative filtering and matrix factorization techniques, along with deep learning-based models like Matrix Factorization Neural Networks (MFNNs), are commonly used in recommendation systems.
4. Game Playing: Linear regression is not applicable for complex game-playing tasks like chess or Go. These games involve large state spaces and require advanced algorithms like Monte Carlo Tree Search (MCTS) or deep reinforcement learning methods such as Deep Q-Networks (DQNs) or AlphaGo.

Conclusion :

Overall, linear regression is a powerful tool for understanding the relationships in data, making predictions, and gaining insights that can be applied in various fields, including economics, social sciences, engineering, and more. However, it's essential to recognize its limitations and use more sophisticated models when the data exhibits complex patterns or relationships.

References:

- https://www.javatpoint.com/linear-regression-in-machine-learning
- https://www.ibm.com/topics/linear-regression
- https://towardsdatascience.com/why-linear-regression-is-not-suitable-for- binary-classification-c64457be8e28