**Applied Machine Learning**

**Lab Report 2**

**Hafiz Ahmad**

**19l-1316**

**Section-8A**

**INTRODUCTION:**

The hypothesis, cost function, and gradient descent aspects of linear regression are the primary focus of this experiment. By fitting a linear equation to the data that was observed, the statistical method known as linear regression is used to model the relationships between variables. This equation is represented by the hypothesis function, and the cost function measures the difference between the predicted and actual values. Gradient descent is an iterative algorithm for minimizing the cost function in order to maximize the parameters of the hypothesis function. Our comprehension of linear regression and its practical applications is enhanced by our comprehension of these ideas.

**OBJECTIVES:**

The objective of this experiment is to gain a comprehensive understanding of how the hypothesis function, cost function, and gradient descent algorithm work collaboratively to optimize the parameters of a linear regression model.

**Procedure:**

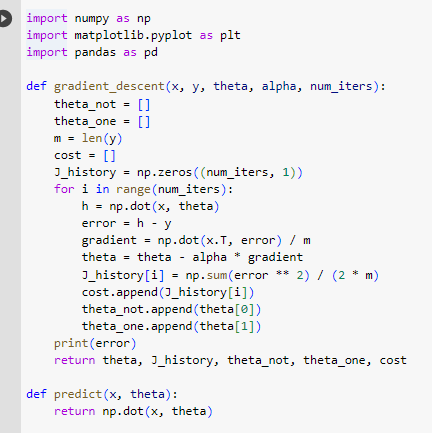
A specific procedure is followed throughout the experiment. To begin, necessary libraries like NumPy, Matplotlib, and pandas are imported for, respectively, data handling, data visualization, and numerical operations. The gradient descent algorithm is then implemented by defining a function called "gradient\_descent." The input variables that this function takes into account are the number of iterations (num\_iters), the learning rate (alpha), the initial parameters (theta), and the input features (x).

Several iterative steps are carried out within the "gradient\_descent" function. In order to store intermediate values, empty lists undergo initializations. The length of the objective qualities (y) is determined, and a cluster to store cost capability values (J\_history) is made. For the specified number of iterations, the gradient descent loop is carried out. During every cycle, anticipated values (h) are determined by performing network augmentation between the information highlights (x) and the boundary vector (theta). Calculated is the difference in error between the actual target values and the predicted values.

The gradient is created by dividing the number of training examples by the multiplied transposed input features (x) and error. The update rule of gradient descent is used to update the parameters (theta), with the learning rate (alpha) determining the update step size. Furthermore, the expense capability an incentive for the ongoing cycle is processed and put away. The lists that keep track of their changes include the current parameter values. After the inclination plummet circle, the last blunder is printed. The final parameter values, the cost function history (J\_history), and the intermediate parameter values are all returned by the "gradient\_descent" function.

Moreover, a "foresee" capability is characterized to compute the anticipated qualities given the information highlights (x) and the learned boundaries (theta). The code's starting point is the main function. It sets the learning rate (alpha), initial parameters (theta), input features (x), target values (y), initial parameters (theta), and number of iterations (num\_iters). The "gradient\_descent" capability is called to get the improved boundaries, cost capability history, and halfway boundary values. The intermediate parameter and cost function values are stored in a DataFrame, which is then saved to a CSV file.

The data points and the hypothesis function are also plotted after the optimized parameters from gradient descent are printed. Also, the expense capability values over cycles are plotted for representation purposes.

 A picture containing text, screenshot, font

Description automatically generated

A screen shot of a computer program

Description automatically generated with low confidence A picture containing text, screenshot, line, diagram

Description automatically generated

A picture containing text, diagram, line, screenshot

Description automatically generated

**Application:**

The ideas shrouded in this analysis, including speculation, cost capability, and angle drop, have boundless applications in AI calculations, especially in straight relapse. These ideas are used in foreseeing stock costs, assessing lodging costs, and breaking down patterns in datasets.

**Issues:**

No issue was found while performing in the lab.

**Conclusion:**

In conclusion, this experiment provided a comprehensive comprehension of the linear regression hypothesis function, cost function, and gradient descent. We were able to successfully improve the parameters of a linear regression model by employing the gradient descent algorithm. The iterative optimization process and the values of the data points, hypothesis function, and cost function were depicted in the code's implementation.