Machine Learning

Assignment

Building Gradient Descent Algorithm from scratch and analyzing the impact of changing learning rate (alpha) and number of epochs against the mean square error

Algorithm Overview:

Gradient Descent algorithm updating the weights by the formula

$$
\begin{aligned}
&\text{repeat until convergence } \{ \\
&\quad \theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\
&\quad \theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot x^{(i)} \\
&\}
\end{aligned}
$$

What we needed:

We needed to monitor how the changes of alpha and changing of epoch number would affect the final value of y prediction against the actual y value using Mean Square Error (mse)

My Approach:

I have realized this pattern in our problem we need to repeat the algorithm many times using different values of alphas and fixed number of epochs or changing the epochs and fixing the value of alpha on all features, so I noticed that it could be solved by 3 nested for loops.

First loop traversing the features , second loop traversing alpha values and finally the final loop traversing the epoch number.

Process:

1. Importing needed libraries
2. Reading the dataset
3. Basic Data Inspection to explore the dataset
4. Scaling the features as features value varies, to be easily plotted them in a graph
5. Creating the function with parameters
    a. List of learning rates
    b. List of Epochs
    c. Boolean is_alpha to be true when measuring the effect of alpha and false when measuring the effect of the epochs
6. Implementing the algorithm
7. Plotting the results
8. Function Calls

Results:

Changing Aplha with fixed number of epochs

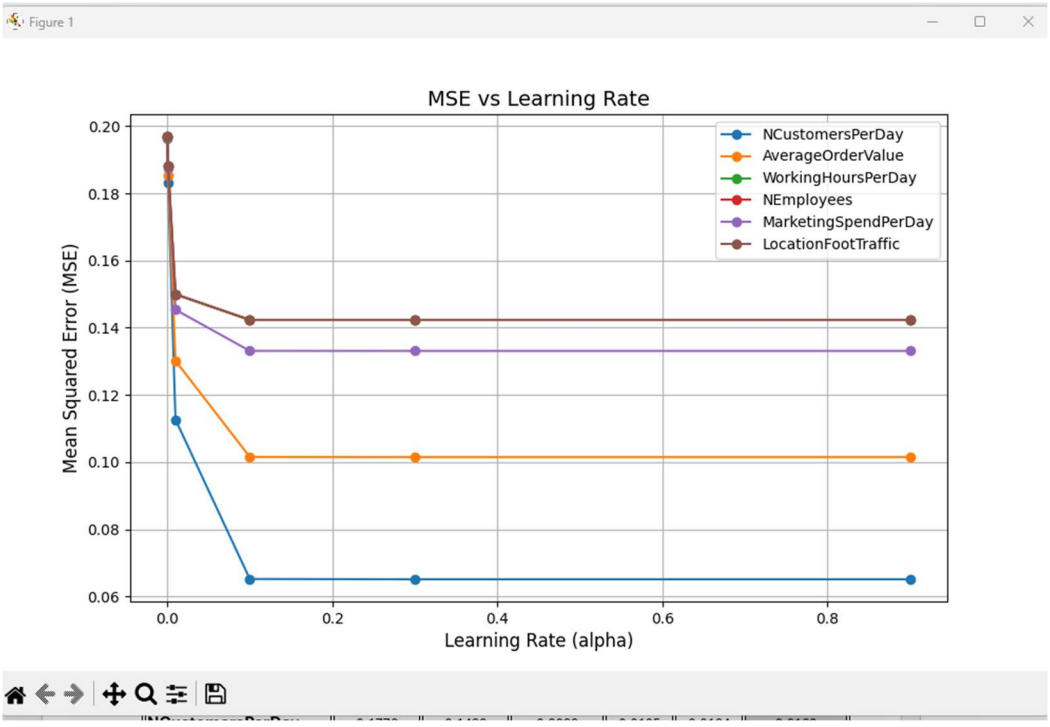
```
linear_regression([0.0001, 0.001, 0.01, 0.1, 0.3, 0.9], [100], True)
```

I have tried different alpha values as shown in the first parameter 0.0001,0.001,0.01,0.1,1,10

I have realized the following

1. Smaller alpha leads to slower converge to the minimum error point
2. Larger alpha could lead to overshooting the minimum error point
3. Increasing Learning Rate above certain threshold leads to increasing mean square error

Graph after scaling the features for better visualization



Optimum Learning Rate Value is 0.9 at fixed 100 epochs as it produces the least error as shown in the table below

Fixed Aplha with changing number of epochs

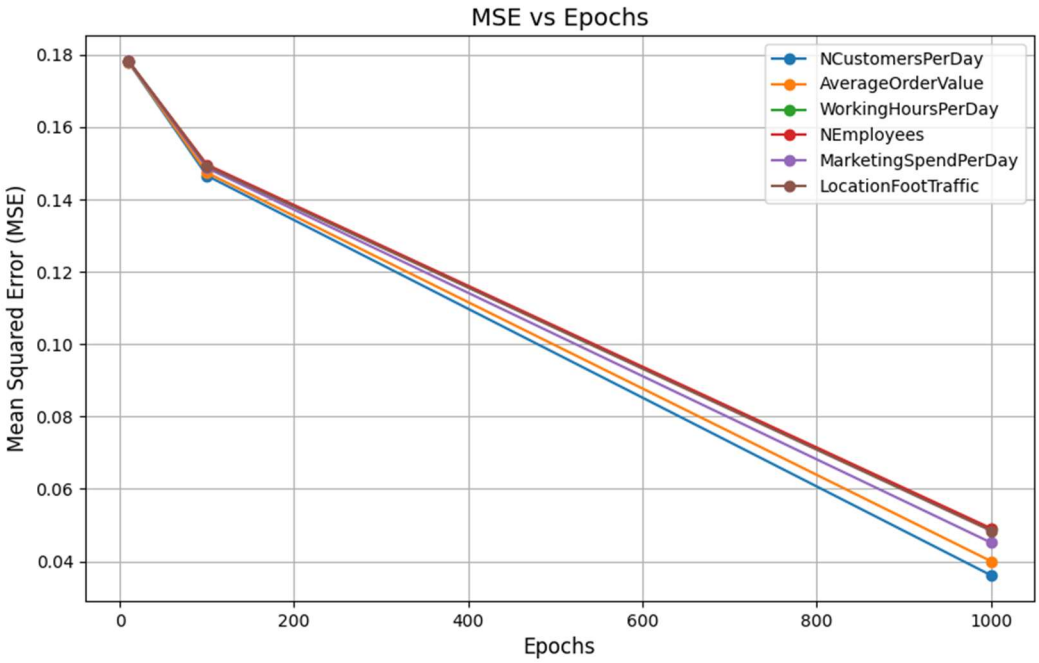
```
linear_regression([0.001], [100,200,1000], False)
```

I have tried different epochs values as shown in the second parameter 10,100,1000

I have realized the following

1. A smaller number of epochs may result in a higher MSE error because the algorithm does not have enough iterations to sufficiently update the values.
2. A very large number of epochs can lead to excessive iterations, causing the algorithm to lose efficiency as updates to the values become negligible.

Graph after scaling the features for better visualization



Optimum Scenario:

Optimum Scenario is to set the epochs with large number and break the algorithm loop if delta error is less than certain threshold (Early Stopping)

MSE vs Learning Rate with fixed 100 epochs

Feature	a = 0.0001	a = 0.001	a = 0.01	a = 0.1	a =0.3	a = 0.9
	MSE Error					
NCustomersPerDay	0.1776	0.1466	0.0362	0.0185	0.0164	0.0162
AverageOrderValue	0.1777	0.1474	0.0401	0.0261	0.0254	0.0253
WorkingHoursPerDay	0.1779	0.1492	0.0489	0.0362	0.0356	0.0355
NEmployees	0.1780	0.1496	0.0492	0.0361	0.0356	0.0356
MarketingSpendPerDay	0.1778	0.1487	0.0453	0.0333	0.0333	0.0333
LocationFootTraffic	0.1779	0.1492	0.0485	0.0361	0.0356	0.0356

MSE vs Epochs with fixed learning rate 0.001

Feature	Epochs = 100	Epochs = 200	Epochs = 1000
	MSE Error		
NCustomersPerDay	0.1192	0.0805	0.0361
AverageOrderValue	0.1206	0.0824	0.0400
WorkingHoursPerDay	0.1240	0.0867	0.0489
NEmployees	0.1246	0.0873	0.0491
MarketingSpendPerDay	0.1229	0.0851	0.0452
LocationFootTraffic	0.1240	0.0865	0.0484

Conclusion:

- **Best Features for Predicting Revenue Per Day:** The most effective features are **Number of Customers Per Day** followed by **Average Order Value**, as they achieve the lowest mean squared error after enough epochs.
- **Best Alpha at 100 Epochs:** The optimal learning rate is **0.9**.
- **Best Number of Epochs: 1000 epochs** yield the lowest MSE and best model performance.
- **Too Small Learning Rate** = Slow Convergence = High Error in small number of epochs
- **Too Large Learning Rate** = Overshooting the Minimum Point
- **Very Large Number of Epochs** = No updates in sum epochs = inefficient algorithm.
- **Very Small Number of Epochs** = High Error Rate, Epochs not enough to reach the minimum point