SCHOOL OF ENGINEERING, TECHNOLOGY & SCIENCE

CSC425: Artificial Intelligence
Section 1

Summer 2023

Assignment 2

Faculty: Amin Ahsan Ali, Ph.D.

Submitted by,

Fahim Shahriar Eram
2022523

Linear Regression

A statistical technique known as linear regression is used in data analysis and machine learning to describe the relationship between a dependent variable (also known as the goal or outcome variable) and one or more independent variables (also known as predictors or features). Finding a linear equation that best captures the relationship between changes in the independent variables and changes in the dependent variable is the aim of linear regression.

The linear equation typically takes the form:

$y_i = \theta_0 + \theta_1 x_i$

Here:

- y_i is the dependent variable.
- x_i is one or more independent variables.
- θ_1 is the coefficient (slope) that represents how the independent variable(s) affect the dependent variable.
- θ_0 is the intercept, which is the value of y_i when x_i is equal to zero.

Finding the values of θ_0 and θ_1 that reduce the difference between the predicted values $\hat{y}_i$ and the actual observed value y_i for the dependent variable is the major goal of linear regression.

Gradient Descent

It is an optimization approach which is frequently used in deep learning and machine learning to reduce the error or cost function related to a model's input parameters. Its main goal is to repeatedly change the model's parameters until the cost function is reduced, or in other words, until it finds the best set of parameters for the task at hand.

This is how Gradient Descent functions:

1. **Initialization**: The parameters are first randomly initialized.
2. **Compute the Gradient**: The gradient of the cost function with respect to the model's parameters is calculated. The cost function's steepest increase is shown by the gradient, which is a vector pointing in that direction. It indicates how much each parameter needs to be adjusted to reduce the cost.
3. **Update Parameters**: By moving a brief distance in the gradient's opposite direction, the parameters are updated. This action, known as the learning rate (α), determines the size of the following step. Typically, the update rule for a parameter θ is expressed as: $\theta^{(1)} = \theta^{(0)} - \alpha \nabla J\left(\theta^{(0)}\right)$. Here, $\nabla J\left(\theta^{(0)}\right)$ represents the gradient of the cost function.
4. **Repeat**: Steps 2 and 3 are repeated iteratively until the cost function converges to a minimum value.

The computation and application of the gradient updates vary depending on the kind of Gradient Descent, such as Batch Gradient Descent, Mini-Batch Gradient Descent, and Stochastic Gradient Descent (SGD). Each has distinct advantages and is appropriate for certain situations.

Batch Gradient Descent is known for its stability and smooth convergence since it considers the entire training dataset in each iteration. However, it can be computationally expensive, especially when dealing with large datasets, as it requires computing the gradients for all data points in each iteration. In each iteration, the parameters are updated using the following equation: $\theta^{(1)} = \theta^{(0)} - \alpha\nabla J\left(\theta^{(0)}\right)$.

Mini-Batch Gradient Descent is an optimization algorithm that strikes a balance between the computational efficiency of Stochastic Gradient Descent (SGD) and the stability of Batch Gradient Descent (BGD). It is more computationally efficient than Batch Gradient Descent because it processes only a subset of the data in each iteration, making it suitable for large datasets. In each iteration, the parameters are updated using the following equation: $\theta^{(1)} = \theta^{(0)} - \alpha\nabla J\left(\theta^{(0)}; minibatch\right)$.

Stochastic Gradient Descent can handle large dataset since it analyses one data point at a time and offers faster updates. It can swiftly adjust to shifting data patterns. Due to the noisy updates, it may be able to avoid local minima. To achieve convergence, the learning rate (α) must be adjusted because noisy updates may make it more difficult for SGD to converge. In each iteration, the parameters are updated using the following equation: $\theta^{(1)} = \theta^{(0)} - \alpha\nabla J\left(\theta^{(0)}; x_i y_i\right)$.

Basis Function

Basis functions are fundamental tools in mathematics and computational sciences used to represent complex functions or vectors in terms of simpler components. They serve as the building blocks for expressing and approximating a wide range of mathematical and computational phenomena. Expressing a function in terms of basis functions involves finding appropriate coefficients that determine the contribution of each basis function to the overall representation.

In this project, an exponential basis function has been used. It can be represented by, $x \mapsto e^{k\varkappa}$. Where, k is a constant (0.75) and x is the dataset containing the features.

Experimental Setup

In this project, Visual Studio Code was used to develop and evaluate machine learning models. Python libraries such as NumPy, Pandas, and Matplotlib are used for computation, handling data, and visualization. Range normalizing, shuffling and splitting dataset are handled by NumPy Array. The ratio of Training Dataset to Testing Dataset is 8:2. Each of the gradient descent variation's function was set up using their appropriate equation. For maintaining consistency, certain parameters were established.

Iteration: Model training iterated until $\| \theta^{(i+1)} - \theta^{(i)} \| \leq$ tolerance where tolerance is 10^{-4}.

Learning Rate: A constant α of 0.001 is chosen.

Batch Size: For Mini Batch Gradient Descent, a batch size of 10 was used.

Plots

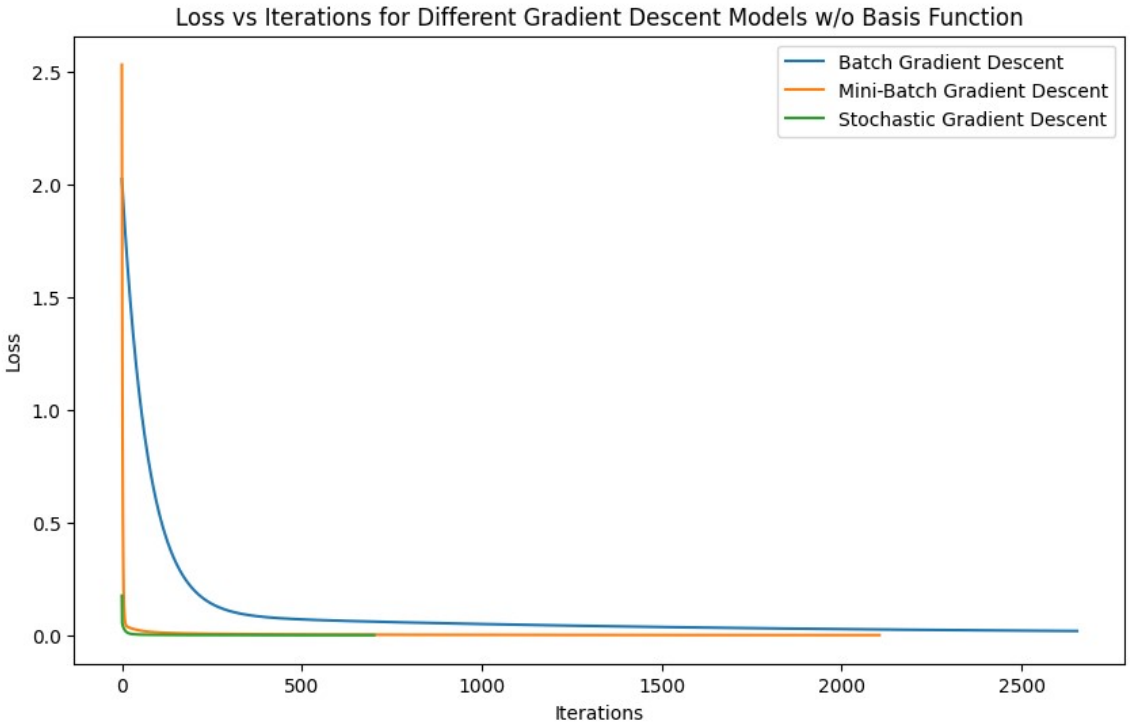

Figure 1: Performance of Gradient Descent Models without Basis Function

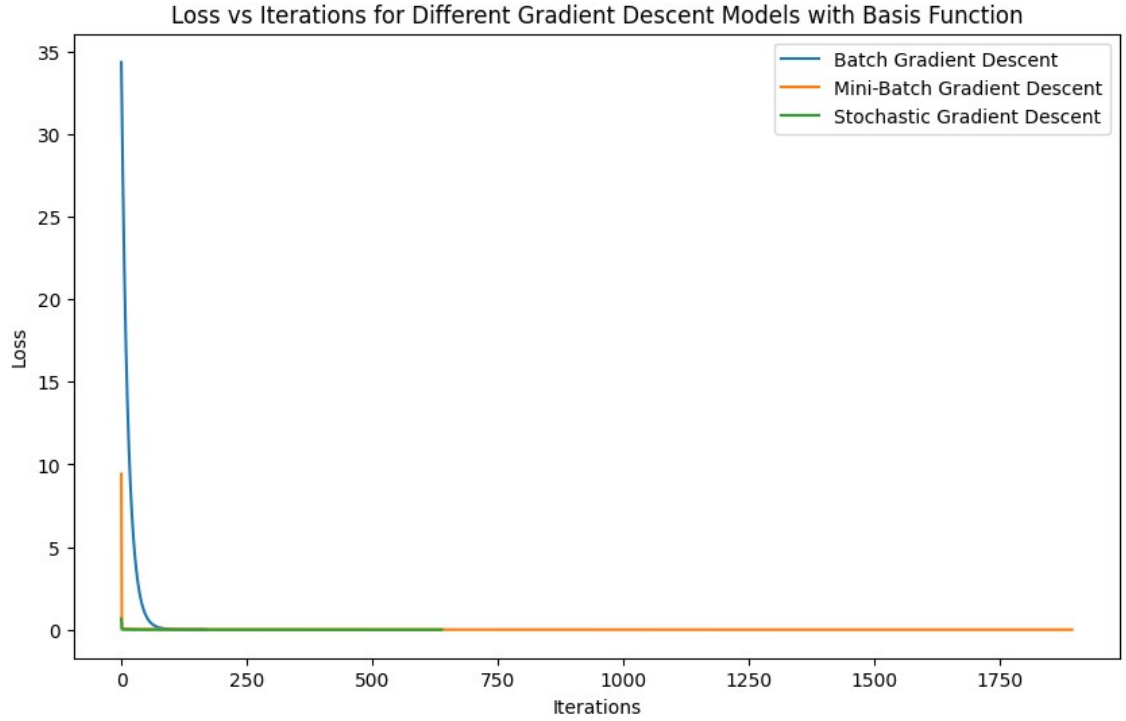

Figure 2: Performance of Gradient Descent Models with Basis Function

In figure 1 we see, when we trained the model without applying the basis function. It took more than 2500 iterations and less than 2750 iterations for BGD, it took more than 2000 iterations and less than 2250 iterations for MBGD, and it took more than 500 iterations and less than 750 iterations for SGD. The initial loss was very low for each model.

Whereas on the other hand, in figure 2 we see, when we trained the model by applying the basis function. The iterations of BGD and MBGD dropped significantly compared to the iterations where we do not apply the basis function. Whereas the iterations of SGB remained very close but the iteration in figure 2 is still lower than the iteration in figure 1. For BGD and MBGD, we also see that the initial loss was significantly higher than compared to the initial loss of BGD and MBGD in figure 1. But the initial loss of SGD remained very low like that in figure 1.

Result

Mean Squared Errors of Gradient Descents without the basis function.

MSE for Test Dataset for BGD: 0.021881350701907935

MSE for Test Dataset for MBGD: 0.0034762438038199285

MSE for Test Dataset for SGD: 0.003291705520112469

The predicted values of the test dataset were very close to the actual values. The value of MSE for BGD was 2% and for MBGD and SGD was 0.3%. Our model was highly accurate; hence the regression performed extremely well.

Feature Importance of Gradient Descents without the basis function.

Batch Features	Weight
GRE Score	-0.159318
TOEFL Score	0.346270
University Rating	0.098878
SOP	0.251365
LOR	0.063326
CGPA	0.210231
Research	0.221951

Mini Batch Features	Weight
GRE Score	0.079616
TOEFL Score	0.180147
University Rating	0.030120
SOP	0.015239
LOR	0.080227
CGPA	0.242509
Research	0.023932

Stochastic Features	Weight
GRE Score	0.124034
TOEFL Score	0.075572
University Rating	0.034185
SOP	-0.003412
LOR	0.078733
CGPA	0.330488
Research	0.020319

Here, for BGD, TOEFL Score proved to be the important feature because it carried the most weight. And, for MBGD and SGD, CGPA proved to be the most important feature because it carried the most weight.

Mean Squared Errors of Gradient Descents using the exponential basis function ($x \mapsto e^{0.75*x}$).

MSE for Test Dataset for BGD: 0.01179344858767355

MSE for Test Dataset for MBGD: 0.0035804568711443434

MSE for Test Dataset for SGD: 0.003482150237586458

Again, the predicted values of the test dataset were extremely close to the actual value. The value of MSE for BGD was 1% and for MBGD and SGD was 0.3%. Our model was yet again highly accurate; hence the regression using a basis function also performed extremely well.

One observation we can see is that the exponential basis function improved the MSE value for BGD model. MSE for BGD improved by (0.021881350701907935 - 0.01179344858767355) / 0.021881350701907935) * 100 = 46% than when it evaluated without applying the basis function.

Feature Importance of Gradient Descents using the basis function.

Batch Features	Weight	Mini Batch Features	Weight	Stochastic Features	Weight
GRE Score	0.130259	GRE Score	0.169549	GRE Score	0.091371
TOEFL Score	0.117743	TOEFL Score	-0.021869	TOEFL Score	0.065136
University Rating	-0.210601	University Rating	0.024218	University Rating	0.029387
SOP	-0.335792	SOP	0.018086	SOP	0.001871
LOR	0.264036	LOR	0.056117	LOR	0.061803
CGPA	0.488789	CGPA	0.300045	CGPA	0.297071
Research	0.049555	Research	0.016265	Research	0.019975

Here, for BGD, MBGD and SGD, CGPA proved to be the most important feature because it carried the most weight.

Earlier we saw, when we do not apply the basis function, the TOEFL Score becomes the most important feature of BGD, and CGPA becomes the most important feature for the other two models. Now we see the actual scenario, why using a basis function is important. Because after using the basis function, we see that CGPA in real was the most important feature for all the models. Therefore, CGPA has the most influence on whether a student will get a chance to get admitted to a university.

Conclusion

Using regression, we evaluated all the types of gradient descent models with and without using a basis function. The MSE for all the models, with and without using the basis function, were extremely low thus proving that our predicted value was exceptionally close to being perfect. The MSE values were immensely accurate.

With or without using the basis function, Stochastic Gradient Descent performed almost identical in terms of loss and iterations. This established how much Stochastic Gradient Descent's robustness and adaptability shine in various scenarios, showcasing its reliability as an optimization algorithm.

The basis function had a big influence when we evaluated the MSE for Batch Gradient Descent, the MSE was improved by 46% than when it was evaluated without using the basis function. This also caused us to find the actual important feature that was influencing the chance of admission. Before applying the basis function, we found out it was TOEFL Score, but after applying the basis function, we found out it was CGPA. CGPA is also the most important feature of other models therefore we can conclude that it is the most influential feature that will affect the chance of admission.