Normalization and Gradient Descent in Linear Regression

Normalization is a crucial step in preparing data for machine learning models. It ensures that all features are on a similar scale, which helps the model learn more effectively and efficiently.

Why Do We Need Normalization?

Imagine you have two features to compare: the size of your house (in square feet) and the number of bedrooms. The size of the house might be 1500 square feet, while the number of bedrooms might be 3. If you try to compare these two numbers directly, the size of the house (1500) is much larger than the number of bedrooms (3). This can make it hard for the computer to understand how important each feature is.

Making Everything the Same Size:

Normalization helps by changing all the numbers so they are around the same size. This way, the computer can compare them more easily and treat both features fairly!

Example:

Let's say you have these numbers for Area and Bedrooms:

- Area: [1500, 1800, 2400]
- Bedrooms: [3, 3, 4]

1. **Calculate the Mean:**
   The mean (average) helps us find the center of the data.

$$X_{\text{mean}} \text{ for Area} = \frac{1500 + 1800 + 2400}{3} = 1900$$

$$X_{\text{mean}} \text{ for Bedrooms} = \frac{3 + 3 + 4}{3} \approx 3.33$$

2. **Calculate the Standard Deviation:**
   The standard deviation (spread) shows how much the values vary from the mean.

$$X_{\text{std}} \text{ for Area} \approx 404$$

$$X_{\text{std}} \text{ for Bedrooms} \approx 0.58$$

3. **Normalize the Numbers:**
   We use the formula:

$$X_{\text{normalized}} = \frac{X - \text{mean}}{\text{std}}$$

   For Area:

$$\frac{1500 - 1900}{404} \approx -0.99$$

$$\frac{1800 - 1900}{404} \approx -0.25$$

$$\frac{2400 - 1900}{404} \approx 1.24$$

For Bedrooms:

$$\frac{3 - 3.33}{0.58} \approx -0.57$$

$$\frac{3 - 3.33}{0.58} \approx -0.57$$

$$\frac{4 - 3.33}{0.58} \approx 1.16$$

Why Do We Do This? Now, all the numbers are around the same size, making it easier for the computer to work with them.

Why Do We Normalize the Target (y)? The same idea applies to the target variable (Price). By normalizing y, we make sure that the predictions the model makes are also on the same scale. This helps the model learn better and faster.

Summary:

- Normalization makes all the numbers similar in size.
- It helps the model compare different features easily.
- It improves the learning process and speeds up training.

By normalizing, the model can learn faster and make better predictions!

Why Do We Add Ones (1) to X?

In linear regression, the formula to predict the price is:

$$\text{Price} = \theta_0 + \theta_1 \times \text{Area} + \theta_2 \times \text{Bedrooms}$$

Here:

- θ_0 is a special number called the intercept (or bias).
- θ_1 and θ_2 are numbers that help the model learn the relationship between price, area, and bedrooms.

To make this work in matrix form, we add a column of ones (1) to X, so the equation becomes:

$$\text{Price} = 1 \times \theta_0 + \text{Area} \times \theta_1 + \text{Bedrooms} \times \theta_2$$

This ensures the model correctly learns θ_0 (the starting point) while adjusting for Area and Bedrooms.

Simple Analogy: Think of θ_0 (theta zero) like a starting point in a race. If you don't add 1, the model doesn't know where to start! By adding 1, you tell the model, "Hey, start from here!" before adjusting for Area and Bedrooms. This helps the model make better predictions by including the intercept (bias) in the learning process.

Gradient Descent Example

Introduction

Gradient descent is an optimization algorithm used to minimize a cost function. In this example, we will use gradient descent to fit a linear regression model to a simple dataset. The goal is to find the best parameters (coefficients) for the model that minimize the difference between the predicted values and the actual values.

Dataset

We have a small dataset with two features (Feature 1 and Feature 2) and one target variable (Target):

Feature 1 (x1)	Feature 2 (x2)	Target (y)
1	2	3
4	5	6
7	8	9

Steps

1. **Initialize Variables:**

   Feature matrix X:

   $$X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 4 & 7 \\ 1 & 5 & 8 \end{bmatrix}$$

   Target vector y:

   $$y = \begin{bmatrix} 3 \\ 6 \\ 9 \end{bmatrix}$$

   Initial model parameters θ:

   $$\theta = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

   Learning rate α:

   $$\alpha = 0.01 \quad \text{(Learning rate, step size for each update)}$$

   Number of iterations:

   $$\text{iterations} = 1000 \quad \text{(Number of times we want to update } \theta\text{)}$$

2. **Add Bias Term:**

   We add a column of ones to the X matrix to account for the bias term (θ_0):

   $$X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 4 & 7 \\ 1 & 5 & 8 \end{bmatrix}$$

   Now X has an extra column that represents the bias term.

3. **Gradient Descent Formula:**

   The gradient_descent function will iteratively update θ using the formula:

   $$\theta = \theta - \frac{\alpha}{m} X^T (X\theta - y)$$

   Where:

   - $m = 3$ is the number of data points.
   - $\alpha = 0.01$ is the learning rate.
   - X is the feature matrix (with the bias column added).
   - y is the target vector.
   - θ are the parameters (weights) we are trying to learn.

4. **Example Calculation:**

   Let's go through one iteration of the gradient descent process.

   Initial Prediction:

   We calculate the initial prediction using the current θ (which is $[0, 0, 0]$ initially):

$$X\theta = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 4 & 7 \\ 1 & 5 & 8 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

   So, the initial prediction is 0 for each data point.

   Error:

   Now we calculate the error (the difference between the predicted value and the actual target values):

$$X\theta - y = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 3 \\ 6 \\ 9 \end{bmatrix} = \begin{bmatrix} -3 \\ -6 \\ -9 \end{bmatrix}$$

   Gradient:

   Next, we calculate the gradient, which tells us how much we need to adjust the parameters:

$$X^T(X\theta - y) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 4 & 7 \\ 1 & 5 & 8 \end{bmatrix}^T \begin{bmatrix} -3 \\ -6 \\ -9 \end{bmatrix}$$

   This will give:

$$\text{gradient} = \begin{bmatrix} -18 \\ -87 \\ -126 \end{bmatrix}$$

   Update Theta:

   Now, we update the values of θ using the gradient and learning rate α:

$$\theta = \theta - \frac{\alpha}{m}\text{gradient}$$

$$\theta = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} - \frac{0.01}{3} \begin{bmatrix} -18 \\ -87 \\ -126 \end{bmatrix}$$

$$\theta = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0.06 \\ 0.29 \\ 0.42 \end{bmatrix}$$

$$\theta = \begin{bmatrix} 0.06 \\ 0.29 \\ 0.42 \end{bmatrix}$$

   After one iteration, θ is updated to $[0.06, 0.29, 0.42]$.

Summary:

- **Gradient Descent:** The function iteratively updates θ to minimize the error (difference between predicted and actual values).
- **Learning Rate (α):** Controls the size of the steps taken towards minimizing the error. If it's too large, we may overshoot; if it's too small, the process will be slow.
- **Iterations:** The number of times the function will update θ. More iterations generally lead to better results.

By running the gradient_descent function for enough iterations, θ will converge to values that minimize the error and make the predictions as accurate as possible.

Making Predictions

Once the model parameters are optimized, you can use them to make predictions on new data points. For example, if you have a new data point with Feature 1 = 3 and Feature 2 = 4, you can predict the target value using the optimized parameters.

Prediction Example

Given Values:

- area = 5000 (square feet)
- bedrooms = 6
- X_{mean} = [2700, 4] (mean of area and bedrooms)
- X_{std} = [890.692614, 0.816496581] (standard deviation of area and bedrooms)
- y_{mean} = 591666.6666666666 (mean of the target variable, price)
- y_{std} = 127202.81268728123 (standard deviation of the target variable, price)
- θ = [2.82814137e-16, 6.95395195e-01, 2.93667900e-01] (model coefficients for the intercept, area, and bedrooms)

Step 1: Standardization of Inputs

We first standardize the inputs (area and bedrooms) using X_{mean} and X_{std}.

Standardized area:

$$\text{standardized area} = \frac{5000 - 2700}{890.692614} = \frac{2300}{890.692614} \approx 2.58$$

Standardized bedrooms:

$$\text{standardized bedrooms} = \frac{6 - 4}{0.816496581} = \frac{2}{0.816496581} \approx 2.45$$

Step 2: Make Prediction

We use the normalized data and multiply it with the corresponding θ values, and add the mean of the target variable:

$$\hat{y} = \theta_0 + \theta_1 \cdot (\text{standardized area}) + \theta_2 \cdot (\text{standardized bedrooms})$$

$$\hat{y} = 2.82814137e - 16 + (6.95395195e - 01 \cdot 2.58) + (2.93667900e - 01 \cdot 2.45)$$

$$\hat{y} \approx 1.79 + 0.72 = 2.51$$

Then, we convert it back to the original scale using the y_{std} and y_{mean}.

$$\text{Predicted Price} = \hat{y} \cdot y_{\text{std}} + y_{\text{mean}} = 2.51 \cdot 127202.81268728123 + 591666.6666666666$$

$$\text{Predicted Price} \approx 318153.57 + 591666.67 = 909820.24$$

Conclusion

The predicted price for a house with an area of 5000 square feet and 6 bedrooms is approximately \$909,820.