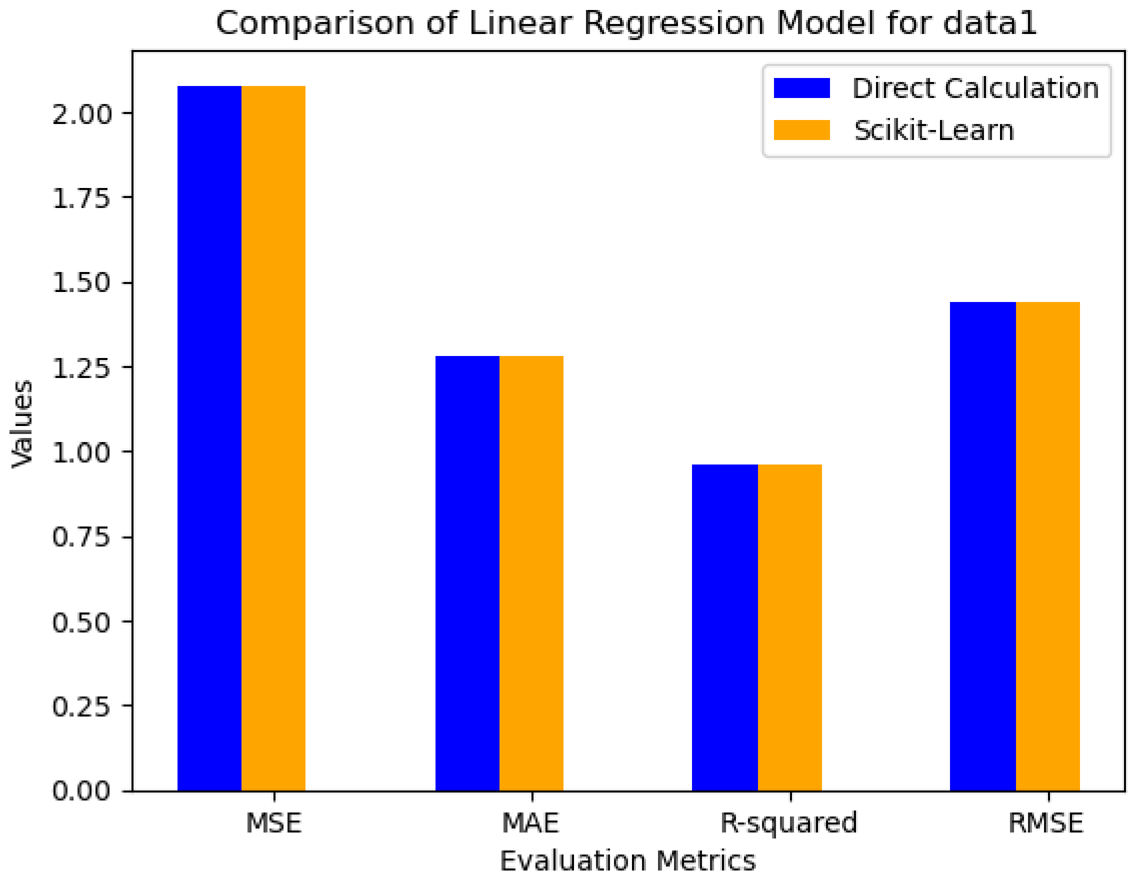
**REPORT**

**Dataset 1:**

| Errors | Normal-Calculation(my model) | Scikit |
|---|---|---|
| **Mean Squared** | 2.078525401777326 | 2.078525401777328 |
| **Mean Absolute** | 1.280555978429146 | 1.2805559784291471 |
| **R-Square** | 0.9579571905586358 | 0.9579571905586357 |
| **Root mean square** | 1.4417091945941547 | 1.4417091945941551 |




**Comparison for weight:**

Weight using gradient descent: [[5.68078713],[ 2.38406007]]

Weight using normal-calculation: [[5.68078713], [2.38406007]]

Observations:

The weight vector using direct calculation and gradient descent are the same, this means both the methods converge to the same solution.

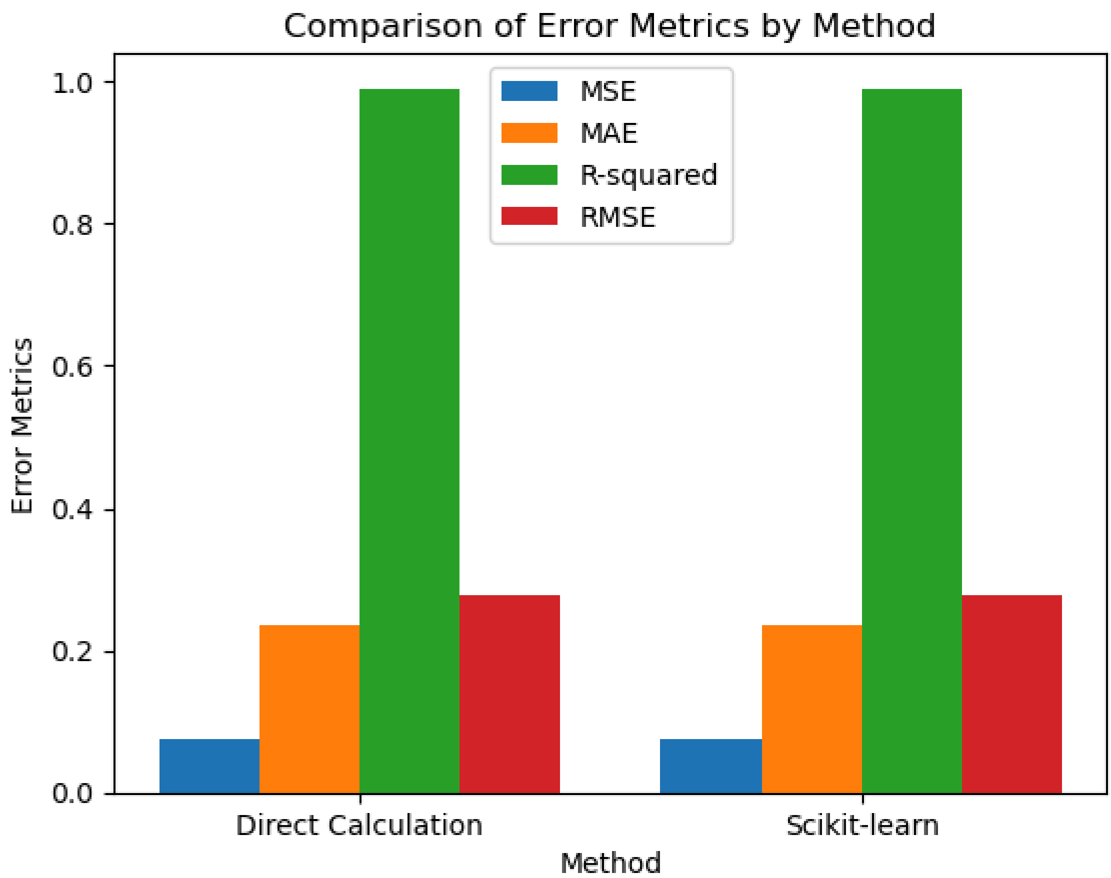
All the three methods of error have negligible differences compared to scikit-learn.

Dataset.R-squared values are close to 1, meaning the model is good and has a strong relationship between the variables.

Since there are only two input features in Dataset 1, linear regression can be directly applied to find the best fit hyperplane. Hyperplane will be straight line.

## Dataset 2:

| Errors | Normal-Calculation | Scikit |
|---|---|---|
| **Mean Squared** | 0.07643342704351964 | 0.07643342704351967 |
| **Mean Absolute** | 0.23498835289025644 | 0.2349883528902572 |
| **R-Square** | 0.9904038522690993 | 0.9904038522690993 |
| **Root mean square** | 0.27646596000867746 | 0.2764659600086775 |



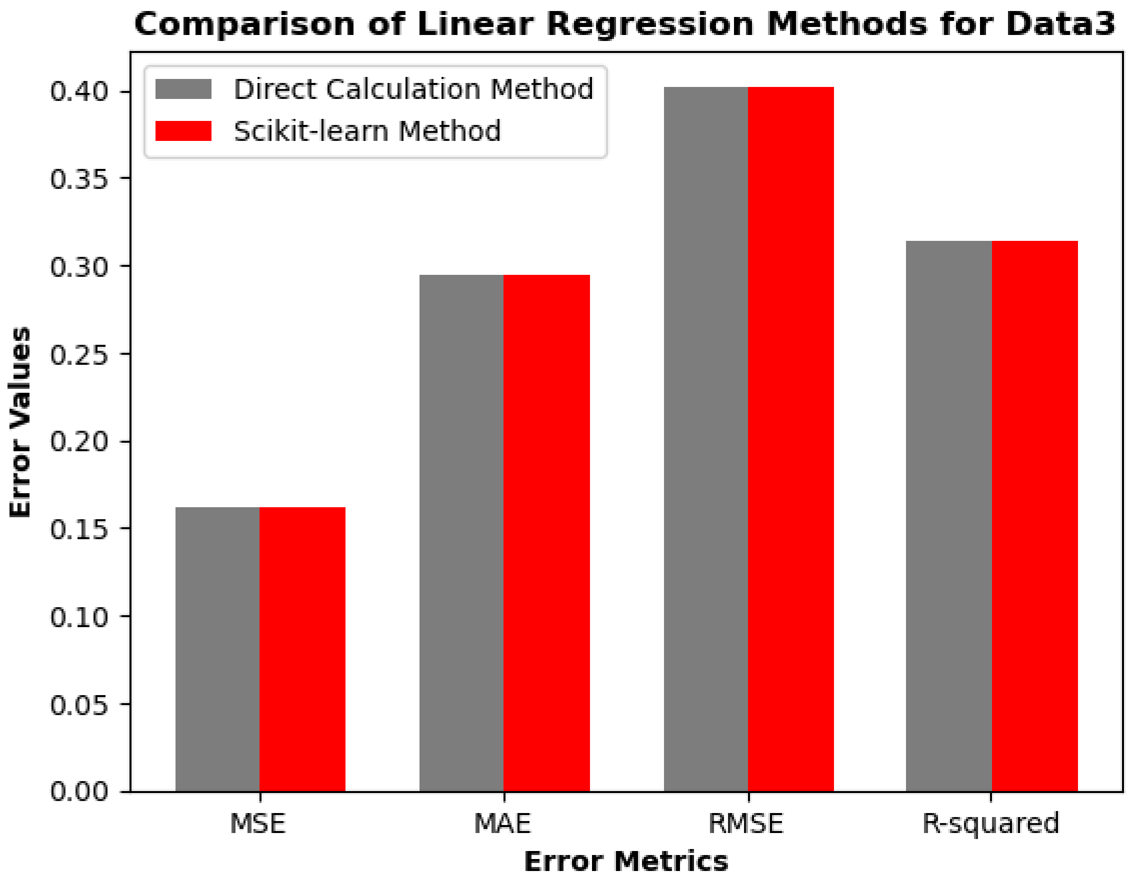

**Comparison for weight:**

Direct Calculation : [[3.68212267], [0.97299745]]
Gradient descent:   [[3.68212267], [0.97299745]]

For dataset2, This requires a non-linear transformation to fit into a linear regression model Because its data points are exponential. I did a non-linear transformation using a logarithmic.

**Dataset 3:**

| Errors | Normal-Calculation | Scikit |
|---|---|---|
| **Mean Squared** | 0.16173044143088555 | 0.16173044143088552 |
| **Mean Absolute** | 0.29467793301310385 | 0.29467793301310374 |
| **R-Square** | 0.40215723471160575 | 0.4021572347116057 |
| **Root mean square** | 0.3136973226728078 | 0.3136973226728079 |



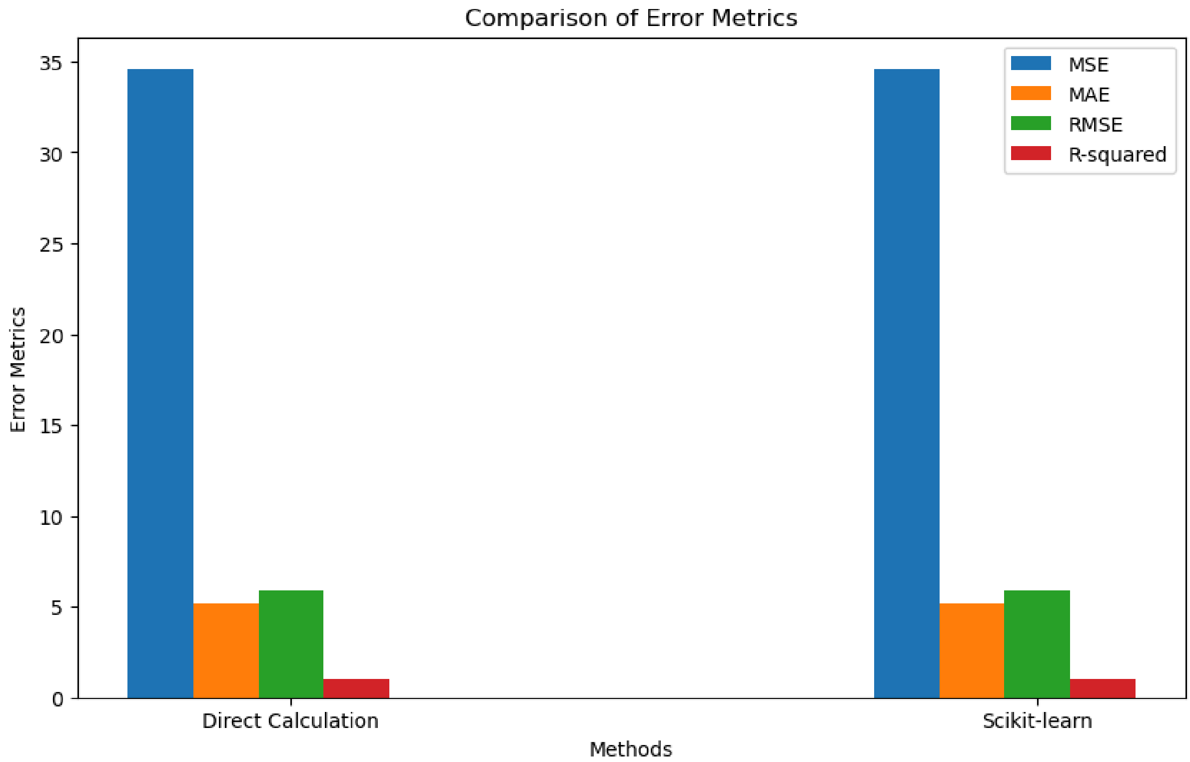

**Comparing weight:**

For direct calculation : [[1.17706208], [0.09419021]]
For gradient descent : [[1.17706208], [0.09419021]]

Direct calculation method and Scikit method gave the same results in terms of error calculation. R-square error is close to 0 and far from 1.our model is not good for dataset 3. I have plotted the graph in my code, there you can clearly see that Linear regression model will not fit for this. Hence no hyperplane.

## Dataset 4 :

| Errors | Direct-Calculation | Scikit-learn |
|---|---|---|
| **Mean Squared** | 34.62048082924356 | 34.62048082924357 |
| **Mean Absolute** | 5.155505630379489 | 5.155505630378744 |
| **R-Square** | 0.9841749058943147 | 0.9841749058943147 |
| **Root mean square** | 5.883917133104745 | 5.883917133104745 |




**Comparing weight:**

Normal Calculation: [[13.23947782] [ 6.13243763] [ 2.39226554] [ 7.74681038]]
Gradient descent: [[13.23947579 6.13243433 2.3922683 7.74681094]]

By looking at 3-4 columns in the dataset, It seems that it may be multiple linear regression. After applying multiple regression, the R-squared value was close to 1. That means my model is good for dataset 4.

**Conclusion:**

Dataset 1 is suitable for standard Linear Regression, Dataset 2 requires a non-linear transformation. Dataset 3 requires a different regression method as linear regression is not applicable on it. Dataset 4 is suitable for standard Linear regression and also multiple linear regression.