# EN3150 Assignment 01

## Learning from Data and Linear Models for Regression

Submitted by:
**P. G. R. S. U. Bandara**
Index Number: **220065A**

Department of Electrical Engineering
University of Moratuwa

August 18, 2025

# Contents

# 1 Linear Regression Impact on Outliers

## 1.1 Task 2: Linear Regression Model and Scatter Plot

Table 1: Data set for linear regression.

| i | $x_i$ | $y_i$ |
|---|---|---|
| 1 | 0 | 20.26 |
| 2 | 1 | 5.61 |
| 3 | 2 | 3.14 |
| 4 | 3 | -30.00 |
| 5 | 4 | -40.00 |
| 6 | 5 | -8.13 |
| 7 | 6 | -11.73 |
| 8 | 7 | -16.08 |
| 9 | 8 | -19.95 |
| 10 | 9 | -24.03 |

The dataset from Table 1 was used to fit a linear regression model, resulting in:

$$y = -3.55727273x + 3.916727272727277$$

The scatter plot of the data points and the regression line is shown below.

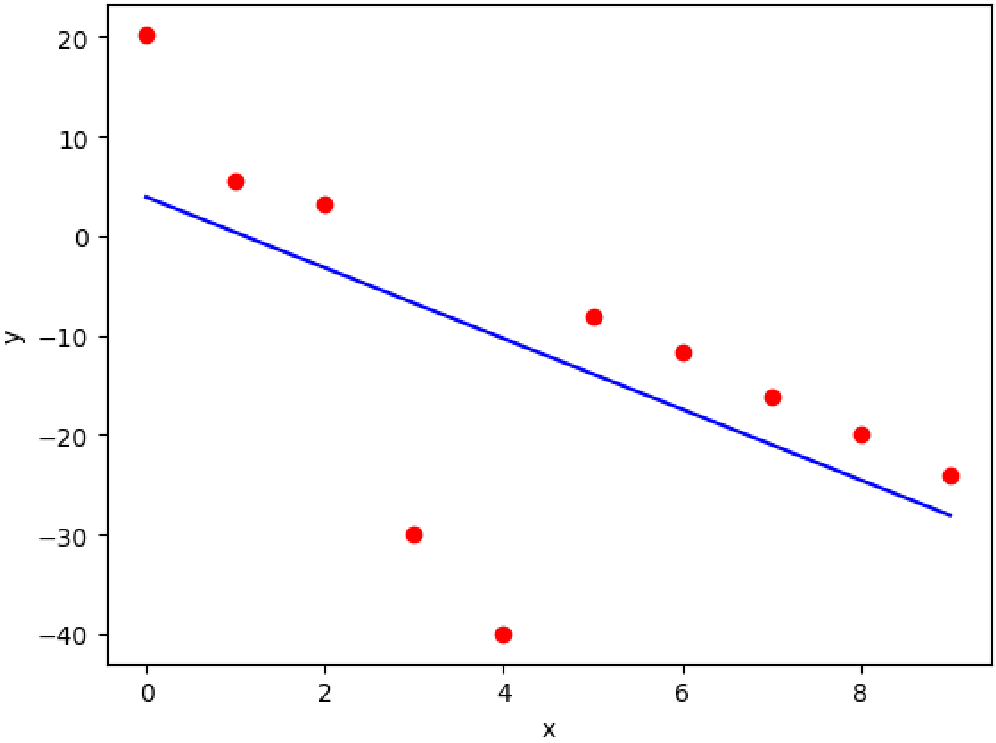

Figure 1: Scatter plot of data points with linear regression model.

## 1.2 Task 4: Loss Function Calculation

- Model 1: $y = -4x + 12$
- Model 2: $y = -3.55x + 3.91$

The loss function values for $\beta = 1, 10^{-6}, 10^{3}$ are:

Table 2: Robust Loss Results for Model 1 and Model 2

| $\beta$ | Model 1 Loss | Model 2 Loss |
|---|---|---|
| 1 | 0.435416 | 0.972847 |
| $1 \times 10^{-6}$ | 1.000000 | 1.000000 |
| 1000 | 0.000227 | 0.000188 |

## 1.3 Task 5: Suitable $\beta$ Value

In regression and model fitting, outliers can heavily distort the results when using ordinary least squares (OLS) estimation, because OLS minimizes the squared residuals and hence gives very high weight to large errors. To overcome this, robust estimation methods are used. These methods modify the loss function in such a way that outliers have less influence on the final model.

$$L(\theta, \beta) = \frac{1}{N} \sum_{i=1}^{N} \rho(r_i), \quad \rho(r) = \frac{r^2}{r^2 + \beta^2} \tag{1}$$

where,

- $r = y - \hat{y}$ is the residual (difference between actual and predicted value).
- $\beta$ is a tuning parameter that determines the boundary between “small” and “large” residuals.

**Role of $\beta$**

- **If $\beta$ is very small (close to 0):** Even small residuals are treated as large relative to $\beta$. This means almost all points are considered as potential outliers. As a result, the model ignores too many useful data points, leading to underfitting.
- **If $\beta$ is very large:** The loss function behaves almost like the ordinary Mean Squared Error (MSE). In this case, the effect of robust estimation disappears, and outliers again dominate the model fitting process.
- **If $\beta$ is chosen close to the scale of normal residual errors:** Then the method effectively distinguishes between normal points and true outliers.
  - For small residuals ($|r| < \beta$): the loss is close to 0, so normal points are properly fitted.
  - For large residuals ($|r| \gg \beta$): the loss saturates close to 1, meaning the outliers are down-weighted and do not distort the model.

**Justification for $\beta = 1$**

A practical and widely accepted choice is $\beta = 1$ (after residual standardization). This value is suitable because it balances robustness and sensitivity, large residuals ($|r| > 1$) are capped, reducing their effect, while small residuals ($|r| < 1$) still provide meaningful gradient information for optimization. Unlike very small $\beta$, it does not incorrectly classify small errors as outliers, thereby preventing the model from ignoring valid points. Moreover, when the residuals are normalized (variance $\approx 1$), $\beta = 1$ corresponds naturally to the typical noise scale in the data. Thus, the model remains accurate for the majority of data points while automatically reducing the influence of extreme outliers, ensuring reliable model fitting.

**Conclusion**

The most suitable value of $\beta$ is $\beta = 1$. This value provides an effective trade-off: it preserves sensitivity to normal data while mitigating the influence of large outliers. Thus, the robust estimator achieves its purpose of fitting the majority of the data without being distorted by extreme values.

**Final Answer:**

$$\beta = 1 \quad \text{is optimal for mitigating outliers, as it balances robustness and accuracy.}$$

## 1.4 Task 6: Most Suitable Model

Using the robust loss function with $\beta = 1$, we obtain:

| **Model** | **Robust Loss $L$** |
|---|---|
| Model 1 ($y = -4x + 12$) | 0.435416 |
| Model 2 ($y = -3.55x + 3.91$) | 0.972847 |

Table 3: Comparison of models using robust loss with $\beta = 1$

**Observation:** Model 1 has a lower robust loss, indicating that it fits the majority of the data better after down-weighting outliers.

Model 1 achieves a robust loss of $L = 0.435416$, which is significantly lower than Model 2 (0.972847), indicating that it fits the majority of the data more accurately. Model 2, being the ordinary least squares (OLS) regression model, minimizes MSE but is strongly influenced by outliers, which pull the line and worsen the fit for the bulk of the data. In contrast, the robust estimator with $\beta = 1$ ignores these outliers, making Model 1 more representative of the main trend. This approach allows Model 1 to maintain sensitivity to normal points while reducing the influence of large residuals, whereas Model 2 shows a higher robust loss due to its susceptibility to outliers.

**Conclusion**

Based on the robust estimator with $\beta = 1$, the most suitable model is Model 1:

$$y = -4x + 12$$

It achieves a lower robust loss, indicating that it fits the majority of the data more accurately. Model 1 mitigates the impact of outliers while still fitting normal residuals effectively. Intuitively, robust estimation prioritizes the "central trend" over outliers, which explains why Model 1 is superior in this scenario.

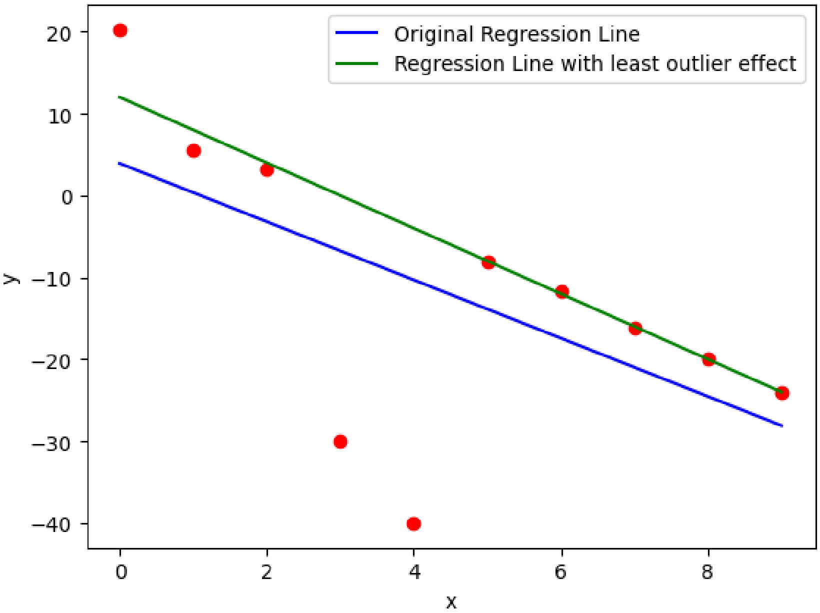

Figure 2: Scatter plot of data points with the original and robust regression lines.

## 1.5 Task 7: Outlier Impact Reduction

In ordinary least squares (OLS), the loss function is

$$\text{RSS} = \sum_i (y_i - \hat{y}_i)^2 \tag{2}$$

where the residuals are

$$r_i = y_i - \hat{y}_i$$

Squaring the residuals makes large errors extremely large, so even a few outliers can "pull" the fitted line significantly.

The robust estimator uses the modified loss function:

$$L(\theta, \beta) = \frac{1}{N} \sum_{i=1}^{N} \rho(r_i), \quad \rho(r) = \frac{r^2}{r^2 + \beta^2} \tag{3}$$

where $\beta$ is a threshold that separates small residuals from large ones.

**How it reduces the impact of outliers:**

Small residuals ($|r| \ll \beta$) have $\rho(r) \approx \frac{r^2}{\beta^2} \ll 1$, meaning the points are treated normally and contribute fully to fitting the model. Large residuals ($|r| \gg \beta$) have $\rho(r) \approx 1$, so the contribution of the point is capped, and outlier errors do not dominate the loss. These points are effectively treated as outliers and down-weighted, allowing the robust estimator to focus on the majority of the data.

**Conclusion**

The robust estimator reduces the influence of outliers while still fitting the majority of the data accurately, unlike ordinary least squares (OLS), which is highly sensitive to a few outliers.

## 1.6 Task 8: Alternative Loss Function

The Huber loss is proposed:

$$L_\delta(r) = \begin{cases} \frac{1}{2}r^2, & \text{if } |r| \leq \delta \\ \delta|r| - \frac{1}{2}\delta^2, & \text{if } |r| > \delta \end{cases}$$

where $r = y_i - \hat{y}_i$ is the residual and $\delta$ is a threshold parameter (play a role like $\beta$)

**Explanation:**

- For small residuals ($|r| \leq \delta$), the loss is quadratic, similar to ordinary least squares (OLS). This ensures that the majority of points contribute fully to fitting the model.
- For large residuals ($|r| > \delta$), the loss becomes linear rather than quadratic, limiting the impact of extreme outliers. This prevents outliers from dominating the loss function.

**Conclusion:** The Huber loss is particularly effective in robust estimation because it balances sensitivity and robustness: it behaves like OLS for inliers, while reducing the influence of outliers. By using the Huber loss, the model focuses on the majority of the data without being unduly affected by outlier errors.

# 2 Loss Function

## 2.1 Task 1: MSE and BCE Loss Values

Table 4: MSE and BCE loss values for predictions with $y = 1$.

| True $y$ | Prediction $\hat{y}$ | MSE | BCE |
|---|---|---|---|
| 1 | 0.005 | 0.990025 | 5.298317 |
| 1 | 0.010 | 0.980100 | 4.605170 |
| 1 | 0.050 | 0.902500 | 2.995732 |
| 1 | 0.100 | 0.810000 | 2.302585 |
| 1 | 0.200 | 0.640000 | 1.609438 |
| 1 | 0.300 | 0.490000 | 1.203973 |
| 1 | 0.400 | 0.360000 | 0.916291 |
| 1 | 0.500 | 0.250000 | 0.693147 |
| 1 | 0.600 | 0.160000 | 0.510826 |
| 1 | 0.700 | 0.090000 | 0.356675 |
| 1 | 0.800 | 0.040000 | 0.223144 |
| 1 | 0.900 | 0.010000 | 0.105361 |
| 1 | 1.000 | 0.000000 | -1.110$\times 10^{-15}$ |

`# Adding a small epsilon` $(= 10^{-15})$ `to avoid` $\log(0)$

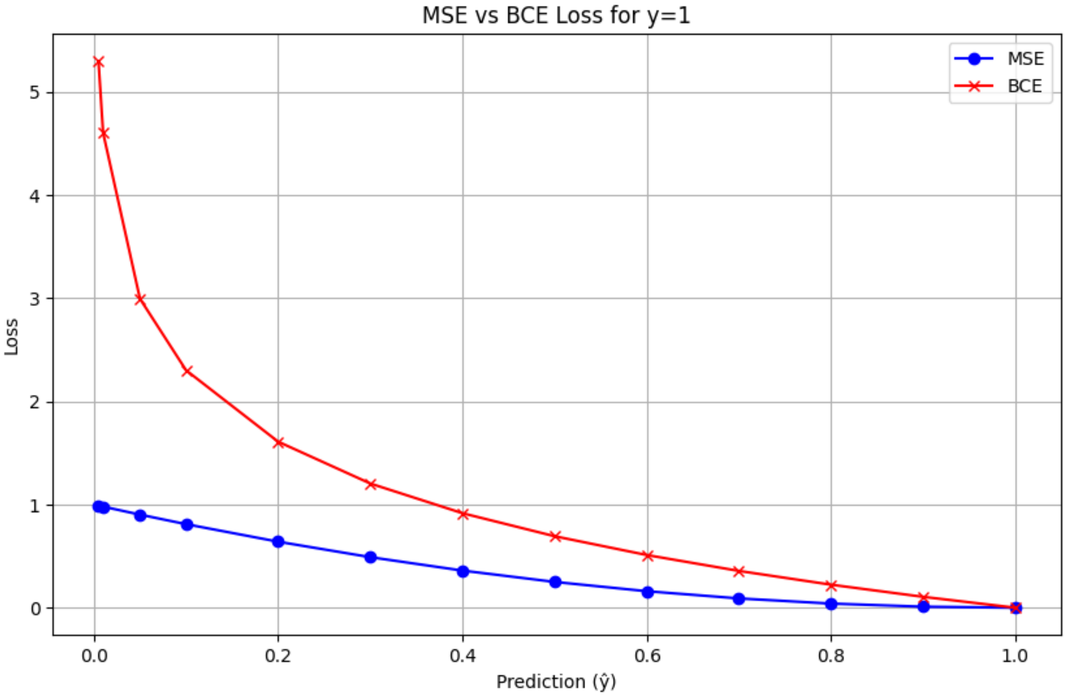

Figure 3: MSE and BCE loss functions for $y = 1$.

### 2.2 Task 2: Loss Function Selection

**Application 1: Mean Squared Error (MSE)-Linear Regression**

**Justification:** MSE calculates the average of the squared differences between the predicted values $\hat{y}$ and the true values $y$. Since the target is continuous, MSE effectively measures how far the predictions are from the actual values and penalizes larger errors more strongly. This makes it ideal for regression problems with continuous outputs.

**Application 2: Binary Cross-Entropy (BCE)-Logistic Regression**

**Justification:** BCE measures how close the predicted probability $\hat{y}$ to the true class label $y$ ($y \in \{0, 1\}$). It penalizes predictions that assign low probability to the true class much more heavily, which is crucial for classification tasks. BCE is therefore ideal for logistic regression, where the output is a probability and the target is binary.

$$\text{If } y = 0, \quad \text{Loss} = -\log(1 - \hat{y}_i)$$

$$\text{If } y = 1, \quad \text{Loss} = -\log(\hat{y}_i)$$

## 3 Data Pre-processing

### 3.1 Task 1: Feature Scaling

**In standard scaling**, the data is centered to have mean 0 and standard deviation 1. This makes it suitable for features that are roughly Gaussian. However, for sparse feature, standard scaling shifts zeros away from zero, which destroys sparsity.

$$z_{\text{scaled}} = \frac{z - \mu}{\sigma}$$

**In Min–Max scaling**, values are mapped to the range $[0, 1]$, preserving relative distances between values. For sparse features, zeros are shifted to a positive value, which can distort the sparse structure.

$$z_{\text{scaled}} = \frac{z - z_{\text{min}}}{z_{\text{max}\,-z_{\text{min}}}}$$

**In Max–Abs scaling**, data is mapped to $[-1, 1]$ while preserving the original sparsity and relative magnitudes. Exact zeros remain zeros, making it ideal for sparse signals or data where zeros have special significance.

$$z_{\text{scaled}} = \frac{z}{\max(|z|)}$$

**Feature 1 (Sparse Signal):** This feature consists mostly of zeros with a few spikes (and one spike modified based on the index). Certain scaling methods are problematic here. Standard scaling shifts zeros to non-zero values, destroying sparsity. Min–Max scaling also shifts zeros to a positive minimum, altering the sparse structure. Max–Abs scaling, on the other hand, preserves zeros, maintains relative spike magnitudes, and keeps the signal sparse. It ensures that the inherent structure of the signal remains intact after scaling. In following plot, a stem plot after max–abs scaling would show zeros remaining exactly at zero, with spikes scaled proportionally.

**Feature 2 (Gaussian Noise):** This feature is continuous and roughly Gaussian, making standard scaling the best choice. It centers the noise to zero mean and scales it to unit variance, preserving the Gaussian shape and making it suitable for most machine learning models. Min–Max or Max–Abs scaling could distort the Gaussian distribution and reduce model effectiveness. In following plot, a stem plot after standard scaling would show the noise centered at zero with normalized variance, retaining its Gaussian shape.

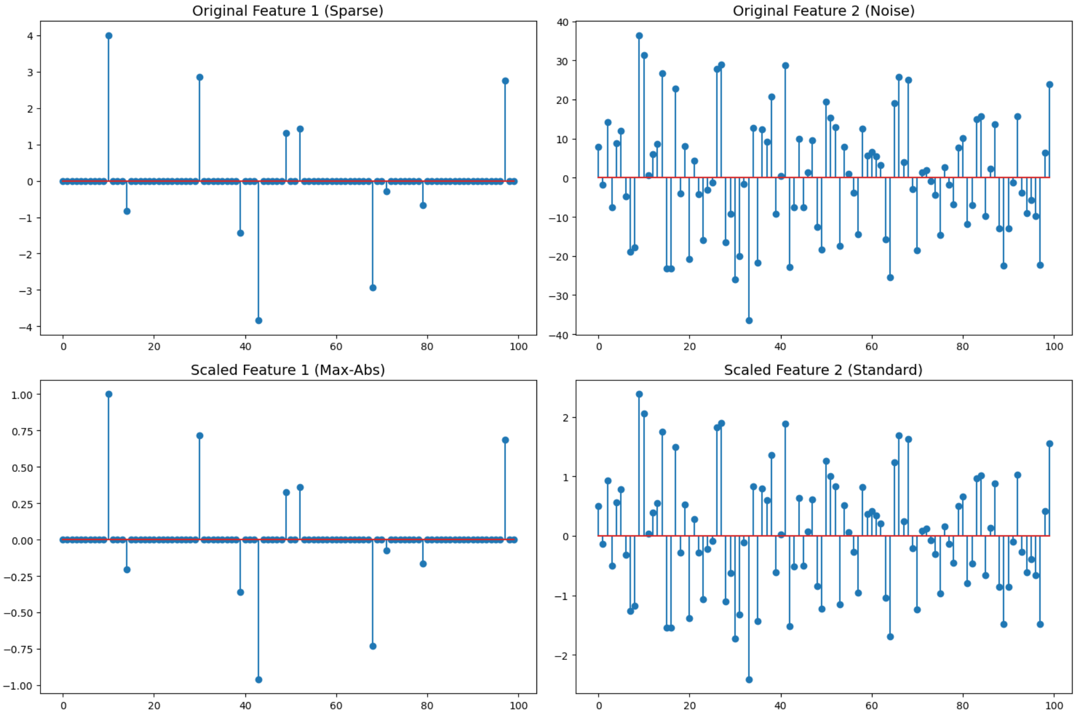

Figure 4: Feature 1 (Sparse signal) and Feature 2 (Gaussian noise).

# 4 References

1. Scikit-learn preprocessing data, `https://scikit-learn.org/stable/modules/preprocessing.html`.

2. Sklearn linear regression, `https://scikit-learn.org/stable/modules/linear_model.html`.

# 5 Appendix

Colab Notebook: https://colab.research.google.com/drive/1gnm98Pe8JcGbsVabbo4w0QuJNCDPkYNB?usp=sharing