Mathematical Equations in the Python and MATLAB Systems for Budget Prediction

Analysis Report

September 9, 2025

Contents

1 Data Preprocessing and Feature Scaling **3**
- 1.1 Log-Transformation of the Target Variable (log1p) . . . 3
  - 1.1.1 Equation . . . 3
  - 1.1.2 Explanation and Parameters . . . 3
  - 1.1.3 Importance and Significance . . . 3
- 1.2 Inverse Log-Transformation (expm1) . . . 3
  - 1.2.1 Equation . . . 3
  - 1.2.2 Explanation and Parameters . . . 3
  - 1.2.3 Importance and Significance . . . 4
- 1.3 Standardization (Standard Scaler / zscore) . . . 4
  - 1.3.1 Equation . . . 4
  - 1.3.2 Explanation and Parameters . . . 4
  - 1.3.3 Importance and Significance . . . 4
- 1.4 Min-Max Scaling . . . 4
  - 1.4.1 Equation . . . 4
  - 1.4.2 Explanation and Parameters . . . 4
  - 1.4.3 Importance and Significance . . . 5

2 Neural Network Components **6**
- 2.1 Rectified Linear Unit (ReLU) Activation Function . . . 6
  - 2.1.1 Equation . . . 6
  - 2.1.2 Explanation and Parameters . . . 6
  - 2.1.3 Importance and Significance . . . 6
- 2.2 Sigmoid Activation Function . . . 6
  - 2.2.1 Equation . . . 6
  - 2.2.2 Explanation and Parameters . . . 6
  - 2.2.3 Importance and Significance . . . 6

3 Loss Function **7**
- 3.1 Mean Squared Error (MSE) . . . 7
  - 3.1.1 Equation . . . 7
  - 3.1.2 Explanation and Parameters . . . 7
  - 3.1.3 Importance and Significance . . . 7

4 Evaluation Metrics **8**
4.1 R-squared (R^2 or Coefficient of Determination) 8
4.1.1 Equation 8
4.1.2 Explanation and Parameters 8
4.1.3 Importance and Significance 8
4.2 Mean Absolute Error (MAE) 8
4.2.1 Equation 8
4.2.2 Explanation and Parameters 8
4.2.3 Importance and Significance 8
4.3 Root Mean Squared Error (RMSE) 9
4.3.1 Equation 9
4.3.2 Explanation and Parameters 9
4.3.3 Importance and Significance 9

1 Data Preprocessing and Feature Scaling

This section details the mathematical transformations applied to the data before it is fed into the neural network. These steps are crucial for stabilizing the model, improving training speed, and handling the inherent statistical properties of the data.

1.1 Log-Transformation of the Target Variable (log1p)

This transformation is applied to the project budget to reduce the effect of a skewed distribution, which is common in financial data.

1.1.1 Equation

$$y' = \ln(1 + y) \tag{1}$$

1.1.2 Explanation and Parameters

- y': The log-transformed value of the target variable (e.g., 'Budget_log').
- y: The original value of the target variable (the project budget).
- ln: The natural logarithm (base e). The '+1' inside the logarithm is crucial, as it handles cases where the budget y could be zero, since ln(0) is undefined. This is known as the "log plus one" or 'log1p' transformation.

1.1.3 Importance and Significance

Financial data like project budgets are often heavily right-skewed (i.e., many low-budget projects and a few very high-budget projects). This skewness can violate the assumptions of many machine learning models and negatively impact their performance. The log-transformation compresses the range of the target variable, making its distribution more symmetric and closer to a normal distribution. This stabilization often leads to a more accurate and robust model that is less influenced by outlier high-budget projects.

1.2 Inverse Log-Transformation (expm1)

After the model makes predictions on the log-transformed scale, this equation is used to revert the predictions back to the original currency scale.

1.2.1 Equation

$$y = e^{y'} - 1 \tag{2}$$

1.2.2 Explanation and Parameters

- y: The final, untransformed prediction in the original units (e.g., PHP).
- y': The model's prediction on the log-transformed scale.
- e: The base of the natural logarithm (Euler's number, ≈ 2.718). This is the mathematical inverse of the 'log1p' function.

1.2.3 Importance and Significance

While the model trains on transformed data, the final results and performance metrics (like MAE and RMSE) must be interpreted in the original monetary units to be meaningful. This inverse transformation is the final step in the prediction pipeline, allowing stakeholders to understand the model's output and its error in real-world terms.

1.3 Standardization (Standard Scaler / zscore)

This scaling technique is applied to the input features (e.g., estimated costs, number of storeys) to ensure they are on a comparable scale.

1.3.1 Equation

$$x' = \frac{x - \mu}{\sigma} \tag{3}$$

1.3.2 Explanation and Parameters

- x': The standardized (scaled) value of a feature.
- x: The original value of a feature for a single project.
- μ: The mean (average) of that feature across all projects in the **training set**.
- σ: The standard deviation of that feature across all projects in the **training set**.

1.3.3 Importance and Significance

Neural networks can be sensitive to the scale of input features. Features with larger numeric ranges (e.g., estimated costs in millions) could dominate the learning process over features with smaller ranges (e.g., number of storeys). Standardization rescales each feature to have a mean of 0 and a standard deviation of 1. This prevents any single feature from having an undue influence on the model's weights during training and often leads to faster convergence of the optimization algorithm (Adam).

1.4 Min-Max Scaling

This scaling technique is applied to the log-transformed target variable ('y_train_scaled') to constrain its values to a fixed range, typically [0, 1].

1.4.1 Equation

$$y'' = \frac{y' - \min(y'_{\text{train}})}{\max(y'_{\text{train}}) - \min(y'_{\text{train}})} \tag{4}$$

1.4.2 Explanation and Parameters

- y'': The min-max scaled value of the log-transformed target.
- y': The log-transformed target value.

- $\min(y'_{\mathbf{train}})$: The minimum value of the log-transformed target observed in the **training set**.
- $\max(y'_{\mathbf{train}})$: The maximum value of the log-transformed target observed in the **training set**.

1.4.3 Importance and Significance

The sigmoid activation function used in the output layer of the Python model constrains its output to the range [0, 1]. To ensure the target values match this range during training, Min-Max scaling is applied. This alignment between the model's output range and the target's range is crucial for effective learning and proper calculation of the loss function. The inverse of this operation is later applied to the model's predictions to bring them back to the log-scale before the final inverse log-transformation.

2 Neural Network Components

This section describes the core mathematical functions inside the neural network that enable it to learn non-linear relationships.

2.1 Rectified Linear Unit (ReLU) Activation Function

ReLU is used as the activation function in the hidden layers of the network.

2.1.1 Equation

$$f(x) = \max(0, x) \tag{5}$$

2.1.2 Explanation and Parameters

- $f(x)$: The output of the ReLU activation.
- x: The input to the neuron (a weighted sum of inputs from the previous layer plus a bias).

2.1.3 Importance and Significance

ReLU is the most popular activation function for deep learning. Its primary role is to introduce non-linearity into the model, allowing it to learn complex patterns that a linear model could not. It is computationally efficient and helps mitigate the "vanishing gradient" problem that can plague other activation functions like sigmoid, leading to faster and more effective training.

2.2 Sigmoid Activation Function

The sigmoid function is used in the output layer of the Python model to bound the prediction.

2.2.1 Equation

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{6}$$

2.2.2 Explanation and Parameters

- $\sigma(x)$: The output of the sigmoid function, which is always in the range (0, 1).
- x: The input to the final neuron.

2.2.3 Importance and Significance

In this regression context, the sigmoid function ensures that the model's final output is a value between 0 and 1. This is specifically chosen to match the [0, 1] range of the Min-Max scaled target variable (‘y_train_scaled‘). This ensures that the model's predictions and the actual targets are on the same scale when the loss is calculated.

3 Loss Function

The loss function quantifies how far the model's prediction is from the actual target. The goal of training is to minimize this value.

3.1 Mean Squared Error (MSE)

MSE is used to measure the error during the model's training phase.

3.1.1 Equation

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{7}$$

3.1.2 Explanation and Parameters

- n: The number of data points in a batch or the entire dataset.
- y_i: The actual (true) scaled target value for the i-th data point.
- $\hat{y}_i$: The model's predicted scaled value for the i-th data point.

3.1.3 Importance and Significance

MSE is the cornerstone of training for many regression models. By squaring the difference between the actual and predicted values, it heavily penalizes larger errors. This encourages the model to avoid making significant mistakes. Furthermore, its mathematical properties (being smooth and differentiable) make it well-suited for gradient-based optimization algorithms like Adam, which adjust the model's weights to minimize this loss.

4 Evaluation Metrics

These metrics are used after training to assess the model's performance on unseen test data, providing a final measure of its accuracy and reliability. All metrics are calculated using the final predictions reverted to their original currency scale.

4.1 R-squared (R^2 or Coefficient of Determination)

R^2 measures the proportion of the variance in the target variable that is predictable from the input features.

4.1.1 Equation

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{8}$$

4.1.2 Explanation and Parameters

- y_i: The actual budget for the i-th project.
- $\hat{y}_i$: The model's predicted budget for the i-th project.
- $\bar{y}$: The mean (average) of all actual budgets in the test set.

4.1.3 Importance and Significance

R^2 provides a scale-free measure of how well the model fits the data. A value close to 1 indicates that the model explains a large portion of the variability in the project budgets. A value of 0 suggests the model is no better than simply predicting the average budget for all projects. It is a key indicator of the model's explanatory power.

4.2 Mean Absolute Error (MAE)

MAE measures the average absolute difference between the predicted and actual values.

4.2.1 Equation

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{9}$$

4.2.2 Explanation and Parameters

- $y_i, \hat{y}_i, n$: Same as in Equation 8.
- $|\ldots|$: The absolute value operator.

4.2.3 Importance and Significance

MAE provides a straightforward and easily interpretable measure of prediction error. It represents the average error in monetary units (e.g., "on average, the model's budget prediction is off by £X"). Because it does not square the errors, it is less sensitive to large outlier errors compared to RMSE.

4.3 Root Mean Squared Error (RMSE)

RMSE is the square root of the MSE and is another measure of the average magnitude of the error.

4.3.1 Equation

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{10}$$

4.3.2 Explanation and Parameters

- $y_i, \hat{y}_i, n$: Same as in Equation 8.

4.3.3 Importance and Significance

Like MAE, RMSE is in the same units as the target variable, making it easy to interpret. However, because it is derived from MSE, it gives a relatively higher weight to large errors. This means RMSE is particularly useful when large prediction errors are especially undesirable. A model with a lower RMSE is generally preferred, as it indicates predictions are closer to the actual values with less variance in its errors.