### **How to implement the CRM**

To implement the Cloud Ratio Model or MLP network, follow these steps:

1. Install necessary libraries: This step installs the required libraries (`pandas`, `matplotlib`, `seaborn`) using pip.

2. Load CSV file: You need to upload your CSV file containing the data. The code uses the `files.upload()` method to upload the file and then reads it using the `pd.read\_csv()` function.

3. Clean and preprocess the data: This step involves cleaning the data by removing any empty white spaces and rounding decimal values. It also converts specific columns to the appropriate data types.

4. Install TensorFlow and import necessary modules: Here, TensorFlow is installed using pip, and the required modules are imported.

5. Split the data into training, validation, and test sets: The dataset is split into three parts: training set (70%), validation set (15%), and test set (15%). This is done using the `sample()` and `drop()` methods on the `Neural\_set` dataset.

6. Extract the labels from the datasets: The labels (dependent variable) are extracted from the respective datasets and stored in separate variables. This is done using the `pop()` method.

7. Normalize the features: The features (independent variables) are normalized to ensure that they are on a similar scale. This is done using the `Normalization` layer from TensorFlow, and the `adapt()` method is used to compute the mean and variance of the training data.

8. Define and compile the model: The neural network model is defined using the `Sequential` API from Keras. It consists of input normalization, two dense layers with ReLU activation, and an output layer. The model is compiled with the Huber loss function and the Adam optimizer.

9. Train the model: The model is trained on the training set using the `fit()` method. The training process runs for 100 epochs, and the validation split is set to 0.2 to monitor the model's performance during training.

10. Evaluate the model on the test set: The trained model is evaluated on the test set using the `evaluate()` method. The results are printed, showing the Huber loss for total radiation.

11. Plot the distribution of prediction errors: The code calculates the prediction errors by subtracting the predicted values from the actual labels. Then, a histogram is plotted to visualize the distribution of these errors.

12. Perform Gaussian Process Regression (optional): This step demonstrates an optional approach using Gaussian Process Regression. It involves randomly sampling 10% of the data, splitting it into training and testing sets, defining the kernel, creating a Gaussian Process model, fitting the model to the data, and making predictions on the test data.

These steps guide you through the process of loading, cleaning, preprocessing, training, evaluating, and analyzing the neural network model using TensorFlow and additional libraries.