



# ***Custom Loss Function***

## **Prepared By:**

Aman Singh Chauhan 23124005

Dipanshu Patidar 23124015

Keshav Aggarwal 23124022

## **Prepared For:**

CSO 211 Lab Report

# ABSTRACT

Loss functions are fundamental to training machine learning models, as they quantify the discrepancy between predicted and actual outcomes, guiding the optimization process. Traditional loss functions, such as Mean Squared Error (MSE) and Cross-Entropy, have been widely used for a variety of tasks, but their effectiveness can be limited in certain complex learning scenarios. These standard functions may struggle to capture intricate patterns in data or provide robust gradients for optimization, especially in cases of imbalanced datasets, noisy labels, or highly non-linear decision boundaries. To address these limitations, a novel loss function is introduced, designed to enhance model performance by incorporating adaptive weighting mechanisms and dynamic sensitivity to gradient variations. This new loss function improves the model's ability to handle difficult optimization landscapes, facilitating better convergence and more accurate predictions across diverse tasks.

We introduce `ankara_mse`, a custom loss function defined above is designed to evaluate the performance of a model predicting time-series data, such as stock prices or other sequential variables, with an emphasis on capturing the directional movements (up or down) in the data. The loss function combines a standard Mean Squared Error (MSE) component with an additional penalty term that specifically addresses the accuracy of the model's predicted price movements in relation to the true price movements.

# Contents

1. Introduction
2. Methodology
  - 2.1 Combination of loss functions
  - 2.2 MSE (Mean Squared Error)
  - 2.3 Directional Movement
  - 2.4 Directional Penalty Mismatch
3. Experiments
  - 3.1 The Apple finance Dataset
    - 3.1.1 Plotting the loss
    - 3.1.2 Metrics on Training Data
    - 3.1.3 Metrics on Test Data
  - 3.2 The CO<sub>2</sub> Emission Dataset
    - 3.2.1 Plotting the loss
    - 3.2.2 Metrics on Training Data
    - 3.2.3 Metrics on Test Data
  - 3.3 Energy Consumption Data
    - 3.3.1 Plotting the loss
    - 3.3.2 Metrics on Training Data
    - 3.3.3 Metrics on Test Data
4. Results

# **1. INTRODUCTION**

Ankara\_mse finds its motivation in the conventional Mean Squared Error loss used in many machine learning models and takes it to new levels . It combines a standard Mean Squared Error (MSE) component with an additional penalty term that specifically addresses the accuracy of the model's predicted price movements in relation to the true price movements. The penalty is applied when the model's predicted direction (i.e., whether the price is predicted to go up or down) does not match the true direction of the price change, and the magnitude of the penalty increases with the severity of the directional mismatch.

By incorporating this directional penalty alongside the traditional MSE, this custom loss function helps train models that not only minimize overall prediction error but also emphasize accurate predictions of price movements, which is particularly useful for financial and time-series forecasting tasks where the direction of change is as important as the magnitude.

## 2. METHODOLOGY

### 2.1 Combination Of Losses

The custom loss function you've implemented combines two components:

1. Mean Squared Error (MSE) loss between the true and predicted prices.
2. Directional Penalty that penalizes the model when the direction of movement (up or down) predicted by the model doesn't match the true direction.

### 2.2 Mean Squared Error (MSE):

The MSE loss is computed between the true values:

$$\text{mse\_loss} = \frac{1}{n} \sum_{i=1}^n (y_{\text{true}} - y_{\text{pred}})^2$$

### 2.3 Directional Movement:

The "movement" or "direction" of the prices for each time step is computed as the difference between today's price and the next day's price. Specifically:

True direction (up or down) for each time step:

$$y_{\text{true\_move}} = \begin{cases} 1 & \text{if } y_{\text{true\_diff}} \geq 0 \\ 0 & \text{if } y_{\text{true\_diff}} < 0 \end{cases} \quad y_{\text{pred\_move}} = \begin{cases} 1 & \text{if } y_{\text{pred\_diff}} \geq 0 \\ 0 & \text{if } y_{\text{pred\_diff}} < 0 \end{cases}$$

If the price movement is positive or zero, it is labeled as "UP" (1); otherwise, it is "DOWN" (0).

## 2.4 Directional Mismatch Penalty:

If the predicted direction does not match the true direction, only then a penalty is applied. Specifically, the penalty is the absolute difference between the true and predicted direction differences, weighted by the mean absolute difference of the true price movements:

$$\text{Custom Loss} = \text{MSE\_loss} + \text{penalty}$$

$$\text{Penalty} = 1_{\text{misaligned}} \cdot |\Delta y_{\text{true},i} - \Delta y_{\text{pred},i}| \cdot \frac{1}{N} \sum_{i=1}^N |\Delta y_{\text{true},i}|$$

## Actual Loss Function

$$\mathcal{L}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}}) = \frac{1}{N} \sum_{i=1}^N (y_{\text{true},i} - y_{\text{pred},i})^2 + 1_{\text{misaligned}} \cdot |\Delta y_{\text{true},i} - \Delta y_{\text{pred},i}| \cdot \frac{1}{N} \sum_{i=1}^N |\Delta y_{\text{true},i}|$$

## 3. EXPERIMENTS

### 3.1 The Apple Stock Finance Dataset

```
1 hsbcc.describe()
```

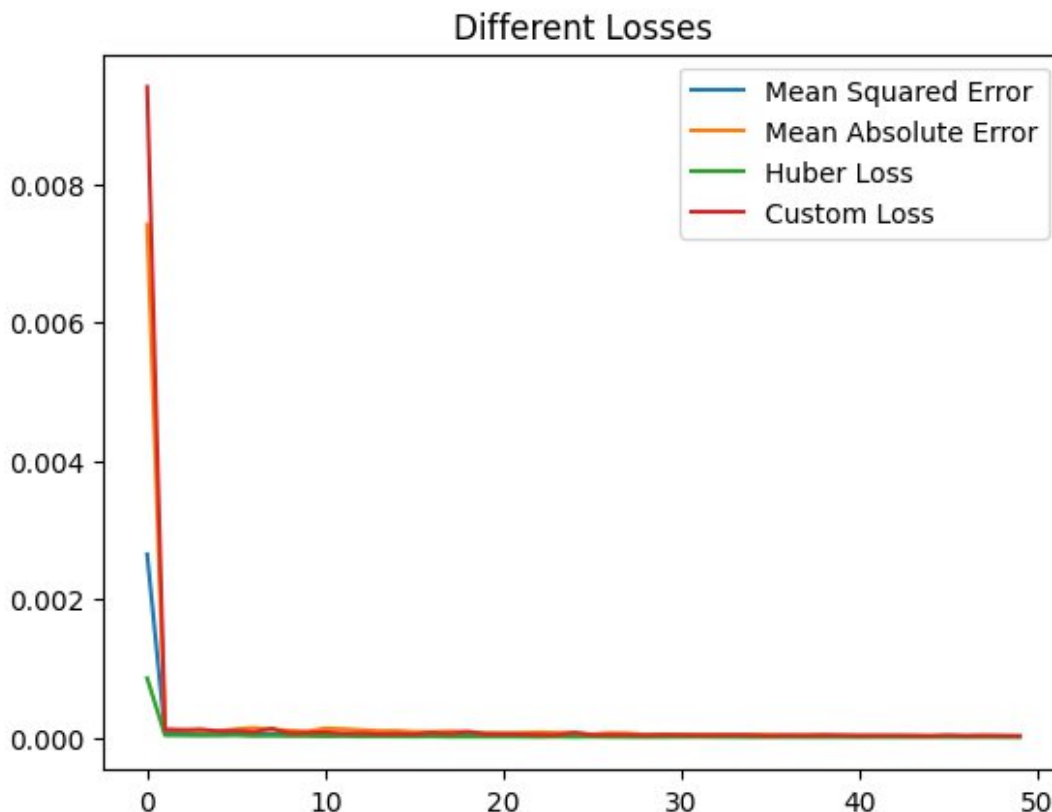
✓ 0.0s

Price	Adj Close	Close	High	Low	Open	Volume
Ticker	AAPL	AAPL	AAPL	AAPL	AAPL	AAPL
count	10957.000000	10957.000000	10957.000000	10957.000000	10957.000000	1.095700e+04
mean	20.747124	21.590338	21.808646	21.354338	21.577175	3.190060e+08
std	44.022975	44.574159	45.014408	44.095707	44.540256	3.357277e+08
min	0.037815	0.049107	0.049665	0.049107	0.049665	0.000000e+00
25%	0.241080	0.296875	0.303571	0.290179	0.296875	1.139152e+08
50%	0.426580	0.524554	0.533482	0.514464	0.522321	2.066456e+08
75%	17.024294	19.714287	19.903214	19.491072	19.785713	3.992352e+08
max	197.144196	198.110001	199.619995	197.000000	198.020004	7.421641e+09

The dataset being used for the first testcase is the AAPL finance dataset containing multiple time based values for data analysis since 1990. We will be working on the Adj close value for the time series analysis.

We will scale the data and pass it to multiple models to get a good grasp on the working of the custom loss function and its performance compared to conventional ones. Each model is run for 50 epochs and with the same training parameters and the graph of loss function is plotted .

#### 3.1.1 Plotting the loss



### 3.1.2 Metrics on Training Data

Loss Used	MSE	MAE	Huber Loss	R2 Score
Mean Squared Loss	$2.376 \times 10^{-5}$	$2.192 \times 10^{-3}$	$1.188 \times 10^{-5}$	0.9996068
Mean Absolute Loss	$2.138 \times 10^{-5}$	$1.905 \times 10^{-3}$	$1.069 \times 10^{-5}$	0.9996538
Huber Loss	$2.070 \times 10^{-5}$	$2.249 \times 10^{-3}$	$1.035 \times 10^{-5}$	0.9996709
Custom Loss	$2.628 \times 10^{-5}$	$1.837 \times 10^{-3}$	$1.314 \times 10^{-5}$	0.9995542

### 3.1.3 Metrics on Test Data

Loss Used	MSE	MAE	Huber Loss	R2 Score
Mean Squared Loss	$1.995 \times 10^{-5}$	$2.135 \times 10^{-3}$	$9.978 \times 10^{-6}$	0.9995217
Mean Absolute Loss	$1.757 \times 10^{-5}$	$1.856 \times 10^{-3}$	$8.786 \times 10^{-6}$	0.9995696
Huber Loss	$1.670 \times 10^{-5}$	$2.193 \times 10^{-3}$	$8.351 \times 10^{-6}$	0.9995834
Custom Loss	$2.262 \times 10^{-5}$	$1.803 \times 10^{-3}$	$1.131 \times 10^{-5}$	0.9994711

Our team has achieved some great efficiency with our custom loss function. The custom loss function has a higher R2 score than all the other 3 losses.



## 3.2 The CO2 Emission Dataset

```
1 data.describe()
```

✓ 0.0s

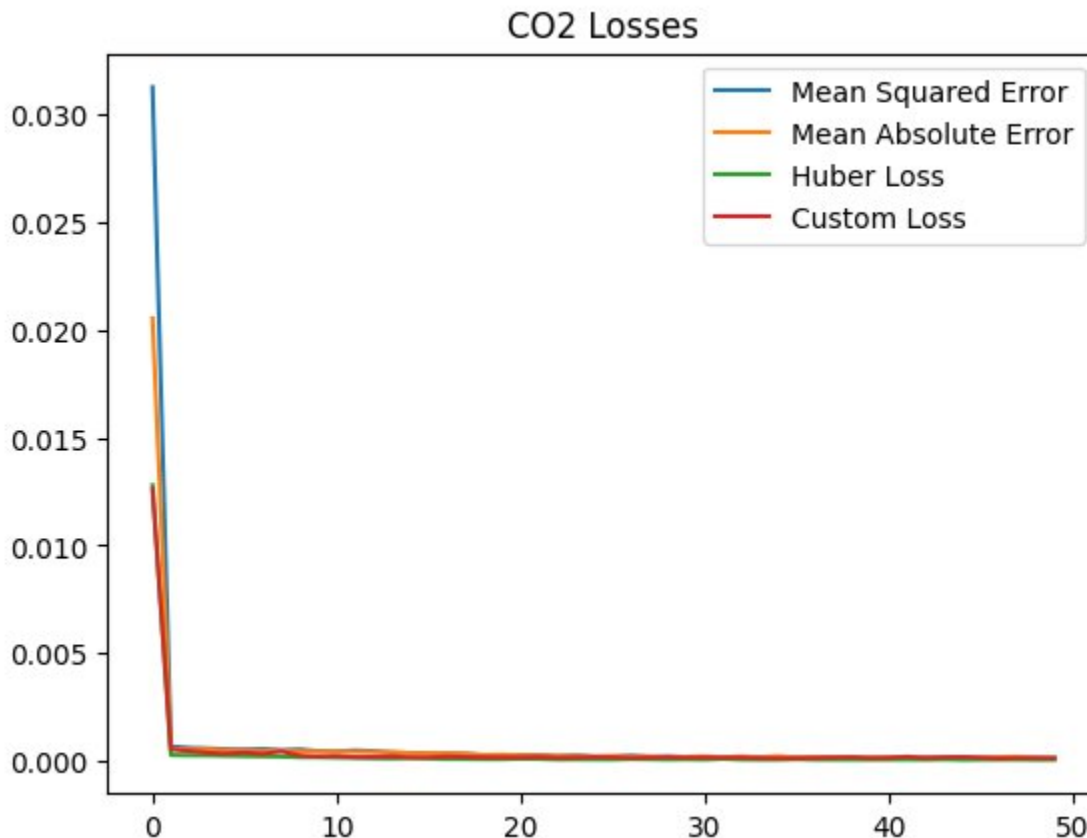
co2

count	2225.000000
mean	340.142247
std	17.003885
min	313.000000
25%	324.800000
50%	338.300000
75%	354.800000
max	373.900000

This dataset is a CO2 emission datasets present in statsmodel api python module. This is a tabular time series dataset with values as early as 1958 and till 2001.

We will scale the data and pass it to multiple models to get a good grasp on the working of the custom loss function and its performance compared to conventional ones. Each model is run for 50 epochs and with the same training parameters and the graph of loss function is plotted .

### 3.2.1 Plotting the loss



### 3.2.2 Metrics on Training Data

Loss Used	MSE	MAE	Huber Loss	R2 Score
Mean Squared Loss	$9.602 \times 10^{-5}$	$7.82 \times 10^{-3}$	$4.801 \times 10^{-5}$	0.998751
Mean Absolute Loss	$14.57 \times 10^{-5}$	$9.62 \times 10^{-3}$	$7.288 \times 10^{-5}$	0.998103
Huber Loss	$7.33 \times 10^{-5}$	$6.78 \times 10^{-3}$	$3.665 \times 10^{-5}$	0.999046
Custom Loss	$10.81 \times 10^{-5}$	$8.12 \times 10^{-3}$	$5.407 \times 10^{-5}$	0.998593

### 3.2.3 Metrics on Test Data

Loss Used	MSE	MAE	Huber Loss	R2 Score
Mean Squared Loss	$9.44 \times 10^{-5}$	$7.98 \times 10^{-3}$	$4.72 \times 10^{-5}$	0.9988264
Mean Absolute Loss	$15.18 \times 10^{-5}$	$10.06 \times 10^{-3}$	$7.59 \times 10^{-5}$	0.9981139
Huber Loss	$7.19 \times 10^{-5}$	$6.94 \times 10^{-3}$	$3.59 \times 10^{-5}$	0.9991062
Custom Loss	$0.01 \times 10^{-5}$	$8.50 \times 10^{-3}$	$5.58 \times 10^{-5}$	0.9986117


This dataset was a bit hard to conquer but as you can see, we have closed the gap on huber loss and defeated the MSE and MAE with our directional analysis.

### 3.3.3 Energy Consumption Data

The PJM Hourly Energy Consumption Data refers to the detailed, time-series data that tracks the electricity consumption (demand) within the service area managed by PJM Interconnection. PJM is a regional transmission organization (RTO) in the United States that coordinates the movement of wholesale electricity across 13 states and the District of Columbia. This includes areas from the Midwest to the mid-Atlantic region. This is a great time series data for our loss function evaluation.

## Hourly Energy Consumption

Over 10 years of hourly energy consumption data from PJM in Megawatts



[Data Card](#)
[Code \(291\)](#)
[Discussion \(13\)](#)
[Suggestions \(0\)](#)

### About Dataset

**PJM Hourly Energy Consumption Data**

PJM Interconnection LLC (PJM) is a regional transmission organization (RTO) in the United States. It is part of the Eastern Interconnection grid operating an electric transmission system serving all or parts of Delaware, Illinois, Indiana, Kentucky, Maryland, Michigan, New Jersey, North Carolina, Ohio, Pennsylvania, Tennessee, Virginia, West Virginia, and the District of Columbia.

The hourly power consumption data comes from PJM's website and are in megawatts (MW).

The regions have changed over the years so data may only appear for certain dates per region.

**Usability** ⓘ  
10.00

**License**  
CC0: Public Domain

**Expected update frequency**  
Never

**Tags**

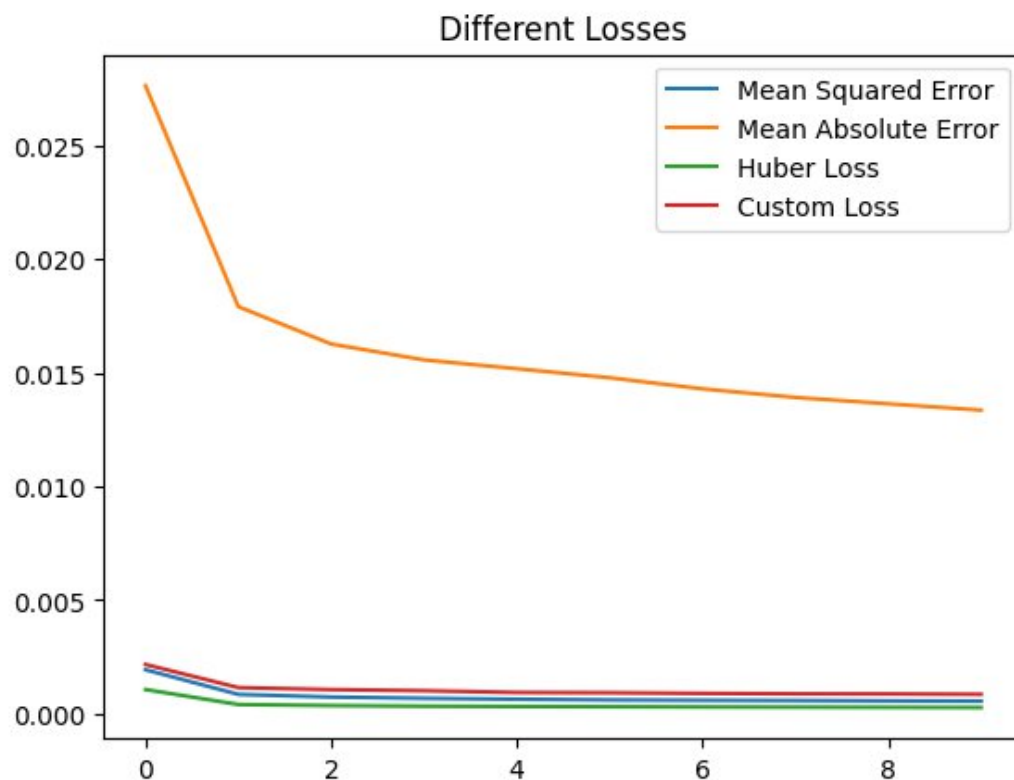
Business
Energy

We will do min max scaling on the data and pass it a stacked LSTM model and run each model for 10 epochs.

This dataset was directly taken from kaggle :

<https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption>

### 3.3.1 Plotting the loss



### 3.3.2 Metrics on Training Data

Loss Used	MSE	MAE	Huber Loss	R2 Score
Mean Squared Loss	$5.673 \times 10^{-4}$	$1.463 \times 10^{-2}$	$2.836 \times 10^{-4}$	0.9780895
Mean Absolute Loss	$5.479 \times 10^{-4}$	$1.369 \times 10^{-2}$	$2.739 \times 10^{-4}$	0.9788387
Huber Loss	$5.191 \times 10^{-4}$	$1.388 \times 10^{-2}$	$2.595 \times 10^{-4}$	0.9799515
Custom Loss	$6.207 \times 10^{-4}$	$1.659 \times 10^{-2}$	$3.103 \times 10^{-4}$	0.9760255

### 3.3.3 Metrics on Test Data

Loss Used	MSE	MAE	Huber Loss	R2 Score
Mean Squared Loss	$5.981 \times 10^{-4}$	$1.475 \times 10^{-2}$	$2.990 \times 10^{-4}$	0.9767582
Mean Absolute Loss	$5.790 \times 10^{-4}$	$1.383 \times 10^{-2}$	$2.895 \times 10^{-4}$	0.9775000
Huber Loss	$5.481 \times 10^{-4}$	$1.394 \times 10^{-2}$	$2.740 \times 10^{-4}$	0.9787040
Custom Loss	$6.509 \times 10^{-4}$	$1.666 \times 10^{-2}$	$3.354 \times 10^{-4}$	0.9747090

As you can see we have further decreased the gap between the huber and our custom loss function. Now the R2 score is roughly same as the huber loss , which is a very efficient loss function.

# Result

In conclusion, the custom loss function developed in this project effectively balances both the accuracy of predictions and the alignment of directional trends. By incorporating a penalty for mismatches in the predicted and true movement directions, it enhances the model's ability to capture not just the magnitude of changes but also the underlying trends.

This approach is particularly valuable for time series forecasting tasks where understanding the direction of change—such as predicting price movements or consumption patterns—is crucial. The combination of Mean Squared Error and directional penalties allows for a more nuanced training process, offering improved performance in scenarios where both accuracy and trend alignment are essential. Further testing and refinement can enhance its robustness and applicability to other domains, making it a versatile tool for complex predictive modeling.