15. Data Science – Machine Learning – Regression Cost functions

Contents

15. Data Science – Machine Learning – Regression Cost functions

1. Cost Functions for Regression

- ✓ In regression, the model predicts an output value for each training data during the training phase.
- ✓ The cost functions for regression are calculated on distance-based error.

2. Types of regression metrics

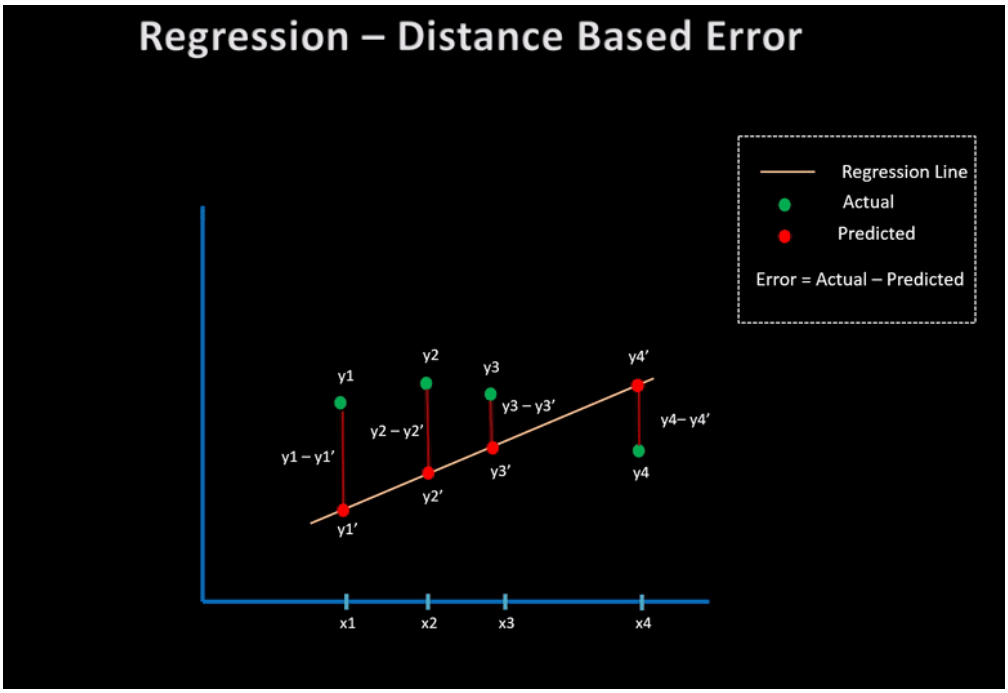
- ✓ There are three metrics in regression,
  - Mean Squared Error (MSE).
  - Root Mean Squared Error (RMSE)
  - Mean Absolute Error (MAE)

3. Distance Based Error

- ✓ Assuming that for a given set of input data, the actual output was **y** and our regression model predicts **y'** then the error in prediction is calculated as

- Error = y - y'

- ✓ This also known as distance-based error and it forms the basis of cost functions that are used in regression models.



4. Mean Squared Error

- ✓ The MSE is calculated as the mean of the squared differences between predicted and expected target values in a dataset.
- ✓ This value never be negative, since we are always squaring the errors.

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

Program Name: Means Squared Error
demo1.py

```
from sklearn.metrics import mean_squared_error

expected = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
predicted = [1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0]

mse_value = mean_squared_error(expected, predicted)

print(mse_value)
```

Output

```
0.35000000000000003
```

5. Root Mean Squared Error

- ✓ The Root Mean Squared Error, or RMSE, is an extension of the mean squared error.
  - RMSE = sqrt(MSE)

Program Name: Root Means Squared Error
demo2.py

```
from sklearn.metrics import mean_squared_error

expected = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
predicted = [1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0]

rmse_value = mean_squared_error(expected, predicted, squared = False)

print(rmse_value)
```

Output

```
0.5916079783099616
```

6. Mean Absolute Error

- ✓ MAE, average absolute difference between predicted and actual values.

Program Name: Means Absolute Error
demo3.py

```
from sklearn.metrics import mean_absolute_error

expected = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
predicted = [1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0]

mae_value = mean_absolute_error(expected, predicted)

print(mae_value)
```

Output

```
0.5
```

7. Formulas

Mean squared error	$\mathrm{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\mathrm{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$