**Loss functions in linear regression**

<https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/>

<https://medium.com/latinxinai/evaluation-metrics-for-regression-models-03f2143ecec2>

<https://medium.com/analytics-vidhya/loss-functions-to-evaluate-regression-models-8dac47e327e2>

**What is Regression?**

Regression is a type of Machine learning that helps in finding the relationship between independent and dependent variables.

In simple words, Regression can be defined as a Machine learning problem where we have to predict continuous values like price, Rating, Fees, etc.

**Why We Require Evaluation Metrics?**

Most beginners and practitioners most of the time do not bother about the model performance. The talk is about building a well-generalized model. A machine learning model cannot have 100% efficiency, else it would be a biased model. This further includes the concept of overfitting and underfitting.

It is necessary to obtain the accuracy on training data, But it is also important to get a genuine and approximate result on unseen data otherwise Model is of no use.

So to build and deploy a generalized model we require to Evaluate the model on different regression model evaluation metrics. These metrics helps us to better optimize the performance, fine-tune it, and obtain a better result.

If one metric is perfect, there is no need for multiple metrics. To understand the benefits and disadvantages of regression evaluation metrics for regression because different evaluation metric fits on a different set of a dataset.

Now, I hope you get the importance of Evaluation metrics. let’s start understanding various regression evaluation metrics used for regression tasks.

***Read this article about***[***Evaluating Regression Models***](https://www.analyticsvidhya.com/blog/2021/10/evaluation-metric-for-regression-models/)

**Dataset**

For demonstrating each evaluation metric using the sci-kit-learn library we will use the placement dataset which is a simple linear dataset that looks something like this.



Now I am applying linear regression on the particular dataset and after that, we will study each evaluation metric and check it on our Linear Regression model.

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

import pandas as pd

cgpa = [6.89, 5.12, 7.82, 7.42, 6.94, 7.89, 6.73, 6.75, 6.09]

package = [3.26, 1.98, 3.25, 3.67, 3.57, 2.99, 2.6, 2.48, 2.31]

df = pd.DataFrame({'cgpa' : cgpa, 'package' : package})

y = df['package']

X = df.drop('package', axis = 1)

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state=2)

lr = LinearRegression()

lr.fit(X\_train,y\_train)

y\_pred = lr.predict(X\_test)

print(y\_pred)

let’s start Exploring various evaluation metrics for regression.

***Also, you can check 12 important Model for***[***Evaluation Metrics for Machine Learning***](https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/)

Types of Regression Metrics

* **Mean Absolute Error(MAE)**
* **Mean Squared Error(MSE)**
* **Root Mean Squared Error(RMSE)**
* **Root Mean Squared Log Error(RMSLE)**
* **R Squared (R2)**
* **R Squared (R2)**

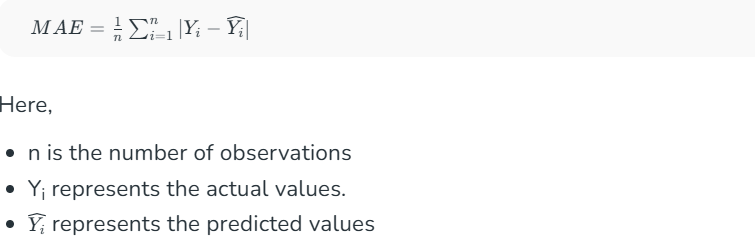
Mean Absolute Error(MAE)

MAE is a very simple metric which calculates the absolute difference between actual and predicted values.

To better understand, let’s take an example you have input data and output data and use Linear Regression, which draws a best-fit line.

Now you have to find the MAE of your model which is basically a mistake made by the model known as an error. Now find the difference between the actual value and predicted value that is an absolute error but we have to find the mean absolute of the complete dataset.

so, sum all the errors and divide them by a total number of observations And this is MAE. And we aim to get a minimum MAE because this is a loss.



Advantages of MAE

* The MAE you get is in the same unit as the output variable.
* It is most Robust to outliers.

Disadvantages of MAE

* The graph of MAE is not differentiable so we have to apply various optimizers like Gradient descent which can be differentiable.

from sklearn.metrics import mean\_absolute\_error

print("MAE",mean\_absolute\_error(y\_test,y\_pred))

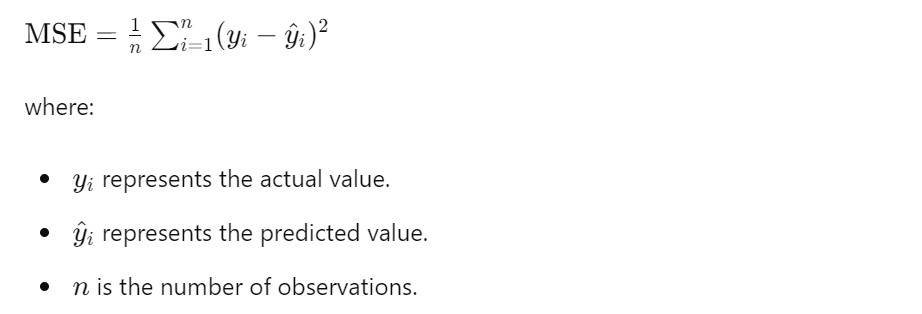
Now to overcome the disadvantage of MAE next metric came as MSE.

**Mean Squared Error(MSE)**

MSE is a most used and very simple metric with a little bit of change in mean absolute error. [Mean**squared error states**](https://www.analyticsvidhya.com/blog/2024/07/mean-squared-error/) that finding the squared difference between actual and predicted value.

So, above we are finding the absolute difference and here we are finding the squared difference.

What actually the MSE represents? It represents the squared distance between actual and predicted values. we perform squared to avoid the cancellation of negative terms and it is the benefit of MSE.



**Advantages of MSE**

* The graph of MSE is differentiable, so you can easily use it as a loss function.

**Disadvantages of MSE**

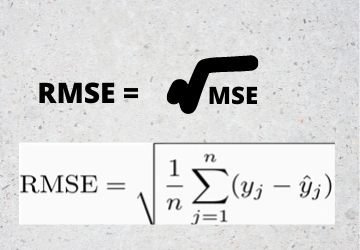
* The value you get after calculating MSE is a squared unit of output. for example, the output variable is in meter(m) then after calculating MSE the output we get is in meter squared.
* If you have outliers in the dataset then it penalizes the outliers most and the calculated MSE is bigger. So, in short, It is not Robust to outliers which were an advantage in MAE.

from sklearn.metrics import mean\_squared\_error

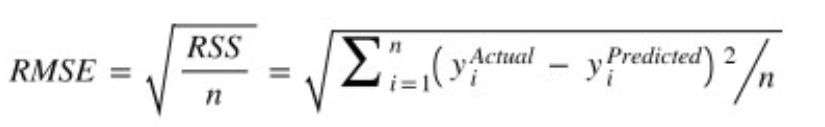
print("MSE",mean\_squared\_error(y\_test,y\_pred))

**Root Mean Squared Error(RMSE)**

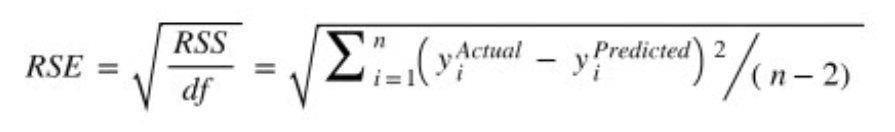
As [**RMSE**](https://www.analyticsvidhya.com/blog/tag/root-mean-square-error/) is clear by the name itself, that it is a simple square root of mean squared error.



The Root Mean Squared Error is the square root of the variance of the residuals. It specifies the absolute fit of the model to the data i.e. how close the observed data points are to the predicted values. Mathematically it can be represented as,



To make this estimate unbiased, one has to divide the sum of the squared residuals by the **degrees of freedom** rather than the total number of data points in the model. This term is then called the **Residual Standard Error(RSE)**. Mathematically it can be represented as,



R-squared is a better measure than RSME. Because the value of Root Mean Squared Error depends on the units of the variables (i.e. it is not a normalized measure), it can change with the change in the unit of the variables.

**Advantages of RMSE**

* The output value you get is in the same unit as the required output variable which makes interpretation of loss easy.

**Disadvantages of RMSE**

* It is not that robust to outliers as compared to MAE.

for performing RMSE we have to NumPy NumPy square root function over MSE.

print("RMSE",np.sqrt(mean\_squared\_error(y\_test,y\_pred)))

Most of the time people use RMSE as an evaluation metric and mostly when you are working with deep learning techniques the most preferred metric is RMSE.

**Root Mean Squared Log Error(RMSLE)**

Taking the log of the RMSE metric slows down the scale of error. The metric is very helpful when you are developing a model without calling the inputs. In that case, the output will vary on a large scale.

* To control this situation of RMSE we take the log of calculated RMSE error and resultant we get as RMSLE.
* To perform RMSLE we have to use the NumPy log function over RMSE.

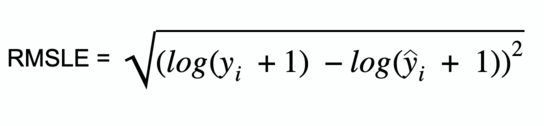
print("RMSE",np.log(np.sqrt(mean\_squared\_error(y\_test,y\_pred))))

It is a very simple metric that is used by most of the datasets hosted for Machine Learning competitions.

**Root Mean Squared Logarithmic Error (RMSLE)**

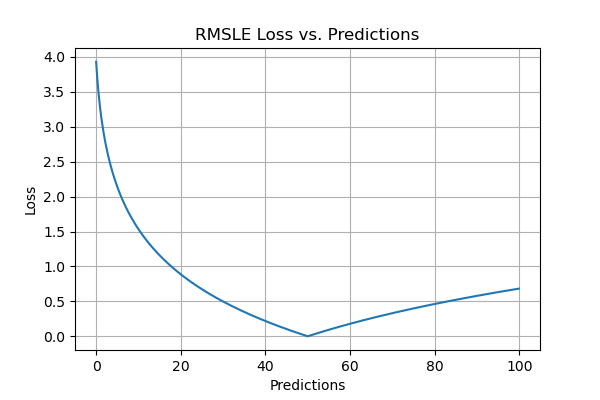
Root Mean Squared Logarithmic Error is calculated by applying log to the actual and the predicted values and then taking their differences. RMSLE is robust to outliers where the small and the large errors are treated evenly.

It penalises the model more if the predicted value is less than the actual value while the model is less penalised if the predicted value is more than the actual value. It does not penalise high errors due to the log. Hence the model has a large penalty for underestimation than overestimation. This can be helpful in situations where we are not bothered by overestimation but underestimation is not acceptable.



Root Mean Squared Logarithmic Error

def root\_mean\_squared\_log\_error(true, pred):  
 square\_error = np.square((np.log(true + 1) - np.log(pred + 1)))  
 mean\_square\_log\_error = np.mean(square\_error)  
 rmsle\_loss = np.sqrt(mean\_square\_log\_error)  
 return rmsle\_loss



RMSLE Plot

**Pros**

* RMSLE is not scale-dependent and is useful across a range of scales.
* It is not affected by large outliers.
* It considers only the relative error between the actual value and the predicted value.

**Cons**

* It has biased penalty where it penalises underestimation more than overestimation.

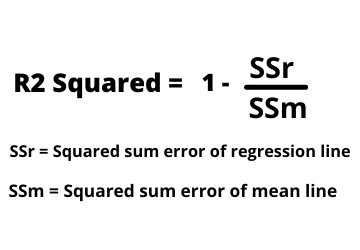
**R Squared (R2)**

R2 score is a metric that tells the performance of your model, not the loss in an absolute sense that how many wells did your model perform.

In contrast, MAE and MSE depend on the context as we have seen whereas the R2 score is independent of context.

So, with help of R squared we have a baseline model to compare a model which none of the other metrics provides. The same we have in classification problems which we call a threshold which is fixed at 0.5. So basically R2 squared calculates how must regression line is better than a mean line.

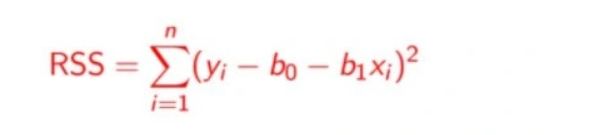
Hence, R2 squared is also known as Coefficient of Determination or sometimes also known as Goodness of fit.



Mathematically it can be represented as,

**R2 = 1 – ( RSS/TSS )**

* **Residual sum of Squares (RSS)** is defined as the sum of squares of the residual for each data point in the plot/data. It is the measure of the difference between the expected and the actual observed output.

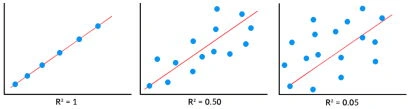


* **Total Sum of Squares (TSS)** is defined as the sum of errors of the data points from the mean of the response variable. Mathematically TSS is,

Total Sum of Squares

where y hat is the mean of the sample data points.

The significance of R-squared is shown by the following figures,



Now, how will you interpret the R2 score? suppose If the R2 score is zero then the above regression line by mean line is equal means 1 so 1-1 is zero. So, in this case, both lines are overlapping means model performance is worst, It is not capable to take advantage of the output column.

Now the second case is when the R2 score is 1, it means when the division term is zero and it will happen when the regression line does not make any mistake, it is perfect. In the real world, it is not possible.

So we can conclude that as our regression line moves towards perfection, R2 score move towards one. And the model performance improves.

The normal case is when the R2 score is between zero and one like 0.8 which means your model is capable to explain 80 per cent of the variance of data.

from sklearn.metrics import r2\_score

r2 = r2\_score(y\_test,y\_pred)

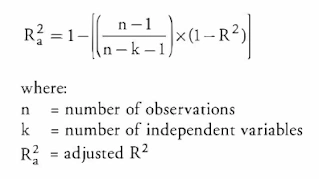
print(r2)

**Adjusted R Squared**

The disadvantage of the R2 score is while adding new features in data the R2 score starts increasing or remains constant but it never decreases because It assumes that while adding more data variance of data increases.

But the problem is when we add an irrelevant feature in the dataset then at that time R2 sometimes starts increasing which is incorrect.

Hence, To control this situation Adjusted R Squared came into existence.



Now as K increases by adding some features so the denominator will decrease, n-1 will remain constant. R2 score will remain constant or will increase slightly so the complete answer will increase and when we subtract this from one then the resultant score will decrease. so this is the case when we add an irrelevant feature in the dataset.

And if we add a relevant feature then the R2 score will increase and 1-R2 will decrease heavily and the denominator will also decrease so the complete term decreases, and on subtracting from one the score increases.

n=40

k=2

adj\_r2\_score = 1 - ((1-r2)\*(n-1)/(n-k-1))

print(adj\_r2\_score)

Hence, this metric becomes one of the most important metrics to use during the evaluation of the model.

Example of Using Regression Metrics on Different Dataset

Here are a few examples of scenarios where you might write about “Using Regression Metrics”:

**1.Predictive Modeling in Real Estate**

You are building a regression model to predict house prices based on features like square footage, number of bedrooms, location, and age of the property. After training the model, you need to evaluate its performance. You can write about how you used regression metrics such as:

* **Mean Absolute Error (MAE)**: To measure the average absolute difference between predicted and actual house prices.
* **Mean Squared Error (MSE)**: To penalize larger errors more heavily, which is useful if you want to avoid significant over- or under-predictions.
* **R-squared (R²)**: To determine how well the model explains the variance in house prices.

You might conclude that while the model has a low MAE, the R² value is moderate, indicating that additional features or more data might be needed to improve the model.

**2.Sales Forecasting for a Retail Business**

You are tasked with predicting monthly sales for a retail store using historical sales data, marketing spend, and seasonal trends. After training a regression model, you evaluate its performance using metrics like:

* **Root Mean Squared Error (RMSE)**: To understand the typical error in your sales predictions in the same units as the sales data.
* **Mean Absolute Percentage Error (MAPE)**: To express the error as a percentage of the actual sales, which is useful for communicating the model’s accuracy to stakeholders.

You might find that the model performs well during non-holiday seasons but struggles during peak shopping periods, indicating a need to incorporate more granular seasonal data.

**3.Energy Consumption Prediction**

You are working on a project to predict energy consumption for a manufacturing plant based on factors like production volume, weather conditions, and time of day. You use regression metrics to assess the model:

* **Explained Variance Score**: To measure how much of the variability in energy consumption is explained by the model.
* **Median Absolute Error**: To evaluate the model’s robustness to outliers, such as unexpected spikes in energy usage.

You might discover that the model performs well overall but struggles to predict extreme values, suggesting the need for outlier detection or a more robust algorithm.

**4.Student Performance Prediction**

You are developing a model to predict students’ final exam scores based on factors like attendance, homework grades, and midterm scores. You use regression metrics to evaluate the model:

* **R-squared (R²)**: To determine how well the model explains the variance in exam scores.
* **Mean Squared Error (MSE)**: To assess the average squared difference between predicted and actual scores.

You might find that the model performs well for students with average scores but struggles to predict high or low performers, indicating a potential need for stratified sampling or additional features.

**5.Stock Price Prediction**

You are building a regression model to predict the future price of a stock based on historical prices, trading volume, and macroeconomic indicators. You evaluate the model using:

* **Mean Absolute Error (MAE)**: To measure the average error in price predictions.
* **Root Mean Squared Error (RMSE)**: To emphasize larger errors, which are critical in financial applications.
* **R-squared (R²)**: To assess how well the model captures the variability in stock prices.

You might conclude that while the model performs reasonably well, the high volatility of stock prices makes it challenging to achieve high accuracy, and you might explore additional features like news sentiment analysis.

**Conclusion**

Evaluating metrics for regression models using appropriate metrics is crucial for assessing their performance and making informed decisions. By understanding and utilizing metrics like MAE, MSE, RMSE, [**R-squared, and others**](https://www.analyticsvidhya.com/blog/2020/07/difference-between-r-squared-and-adjusted-r-squared/), data scientists can quantify the accuracy, goodness of fit, and overall effectiveness of their models. Ultimately, these regression evaluation metrics serve as valuable tools for model selection, optimization, and deployment in real-world regression problems.