3. What are the different loss functions for regression ?

A loss function takes a theoretical proposition to a practical one. Building a highly accurate predictor requires constant iteration of the problem through questioning, modelling the problem with the chosen approach and testing.

The only criteria by which a statistical model is scrutinized is its performance - how accurate the model's decisions are. This calls for a way to measure how far a particular iteration of the model is from the actual values. This is where loss functions come into play.

Loss functions measure how far an estimated value is from its true value. A loss function maps decisions to their associated costs. Loss functions are not fixed, they change depending on the task in hand and the goal to be met.

Loss functions for regression

Regression involves predicting a specific value that is continuous in nature. Estimating the price of a house or predicting stock prices are examples of regression because one works towards building a model that would predict a real-valued quantity.

Let's take a look at some loss functions which can be used for regression problems and try to draw comparisons among them.

Mean Absolute Error (MAE)

Mean Absolute Error (also called L1 loss) is one of the most simple yet robust loss functions used for regression models.

Regression problems may have variables that are not strictly Gaussian in nature due to the presence of outliers (values that are very different from the rest of the data). Mean Absolute Error would be an ideal option in such cases because it does not take into account the direction of the outliers (unrealistically high positive or negative values).

As the name suggests, MAE takes the average sum of the absolute differences between the actual and the predicted values. For a data point x_i and its predicted value y_i, n being the total number of data points in the dataset, the mean absolute error is defined as:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

Mean Squared Error (MSE)

Mean Squared Error (also called L2 loss) is almost every data scientist’s preference when it comes to loss functions for regression. This is because most variables can be modeled into a Gaussian distribution.

Mean Squared Error is the average of the squared differences between the actual and the predicted values. For a data point Y_i and its predicted value $\hat{Y}_i$, where n is the total number of data points in the dataset, the mean squared error is defined as:

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

Mean Bias Error (MBE)

Mean Bias Error is used to calculate the average bias in the model. Bias, in a nutshell, is overestimating or underestimating a parameter. Corrective measures can be taken to reduce the bias post-evaluating the model using MBE.

Mean Bias Error takes the actual difference between the target and the predicted value, and not the absolute difference. One has to be cautious as the positive and the negative errors could cancel each other out, which is why it is one of the lesser-used loss functions.

The formula of Mean Bias Error is:

$$MBE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{n}$$

Where y_i is the true value, $\hat{y}_i$ is the predicted value and ’n’ is the total number of data points in the dataset.

Mean Squared Logarithmic Error (MSLE)

Sometimes, one may not want to penalize the model too much for predicting unscaled quantities directly. Relaxing the penalty on huge differences can be done with the help of Mean Squared Logarithmic Error.

Calculating the Mean Squared Logarithmic Error is the same as Mean Squared Error, except the natural logarithm of the predicted values is used rather than the actual values.

$$MSLE = \frac{1}{n}\sum_{i=1}^{n}\left(\log(Y_i) - \log\left(\hat{Y}_i\right)\right)^2$$

Where y_i is the true value, $\hat{y}_i$ is the predicted value and ’n’ is the total number of data points in the dataset.

Huber Loss

A comparison between L1 and L2 loss yields the following results:

1. L1 loss is more robust than its counterpart.

On taking a closer look at the formulas, one can observe that if the difference between the predicted and the actual value is high, L2 loss magnifies the effect when compared to L1. Since L2 succumbs to outliers, L1 loss function is the more robust loss function.

2. L1 loss is less stable than L2 loss.

Since L1 loss deals with the difference in distances, a small horizontal change can lead to the regression line jumping a large amount. Such an effect taking place across multiple iterations would lead to a significant change in the slope between iterations.

On the other hand, MSE ensures the regression line moves lightly for a small adjustment in the data point.

Huber Loss combines the robustness of L1 with the stability of L2, essentially the best of L1 and L2 losses. For huge errors, it is linear and for small errors, it is quadratic in nature.

Huber Loss is characterized by the parameter delta (δ). For a prediction f(x) of the data point y, with the characterizing parameter δ, Huber Loss is formulated as:

$$L_\delta = \begin{cases} \overbrace{\frac{1}{2}\left(y - f(x)\right)^2}^{\text{Quadratic}}, & if\ |y - f(x)| \leq \delta \\ \underbrace{\delta|y - f(x)|}_{\text{Linear}} - \frac{1}{2}\delta^2, & otherwise \end{cases}$$

Loss functions for classification

Classification problems involve predicting a discrete class output. It involves dividing the dataset into different and unique classes based on different parameters so that a new and unseen record can be put into one of the classes.

A mail can be classified as a spam or not a spam and a person’s dietary preferences can be put in one of three categories - vegetarian, non-vegetarian and vegan. Let’s take a look at loss functions that can be used for classification problems.

Binary Cross Entropy Loss

This is the most common loss function used for classification problems that have two classes. The word “entropy”, seemingly out-of-place, has a statistical interpretation.

Entropy is the measure of randomness in the information being processed, and cross entropy is a measure of the difference of the randomness between two random variables.

If the divergence of the predicted probability from the actual label increases, the cross-entropy loss increases. Going by this, predicting a probability of .011 when the actual observation label is 1 would result in a high loss value. In an ideal situation, a “perfect” model would have a log loss of 0. Looking at the loss function would make things even clearer -

$$J = -\sum_{i=1}^{N} y_i \log(h_\theta(x_i)) + (1 - y_i)\log(1 - h_\theta(x_i))$$

Where y_i is the true label and $h_\theta(x_i)$ is the predicted value post hypothesis.

Since binary classification means the classes take either 0 or 1, if $y_i = 0$, that term ceases to exist and if $y_i = 1$, the $(1-y_i)$ term becomes 0.

Pretty clever, isn’t it?

Categorical Cross Entropy Loss

Categorical Cross Entropy loss is essentially Binary Cross Entropy Loss expanded to multiple classes. One requirement when categorical cross entropy loss function is used is that the labels should be one-hot encoded.

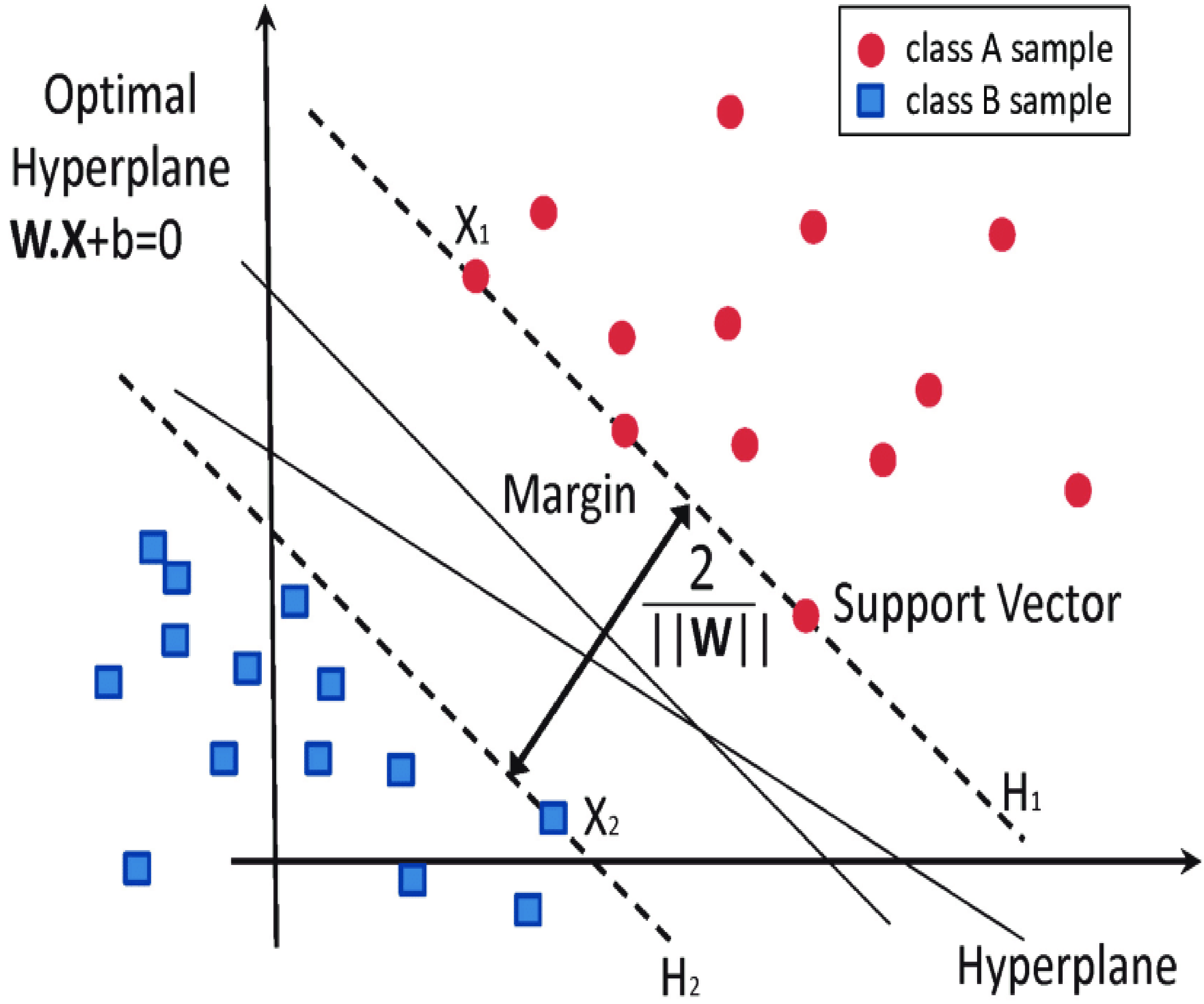
This way, only one element will be non-zero as other elements in the vector would be multiplied by zero. This property is extended to an activation function called softmax, more of which can be found in this article.

Hinge Loss

Another commonly used loss function for classification is the hinge loss. Hinge loss is primarily developed for support vector machines for calculating the maximum margin from the hyperplane to the classes.

Loss functions penalize wrong predictions and does not do so for the right predictions. So, the score of the target label should be greater than the sum of all the incorrect labels by a margin of (at the least) one.

This margin is the maximum margin from the hyperplane to the data points, which is why hinge loss is preferred for SVMs. The following image clears the air on what a hyperplane and maximum margin is:



The mathematical formulation of hinge loss is as follows:

$$SVMLoss = \sum_{j \neq y_i} max(0, s_j - s_{y_i} + 1)$$

Where s_j is the true value and s_{yi} is the predicted value.

Hinge Loss is also extended to Squared Hinge Loss Error and Categorical Hinge Loss Error.

Kullback Leibler Divergence Loss (KL Loss)

Kullback Leibler Divergence Loss is a measure of how a distribution varies from a reference distribution (or a baseline distribution). A Kullback Leibler Divergence Loss of zero means that both the probability distributions are identical.

The number of information lost in the predicted distribution is used as a measure. The KL Divergence of a distribution P(x) from Q(x) is given by:

$$D_{KL}(P||Q) = \begin{cases} -\sum_{x} P(x).log\frac{Q(x)}{P(x)} = \sum_{x} P(x).log\frac{P(x)}{Q(x)}, & \textit{for discrete distributions} \\ -\int P(x).log\frac{Q(x)}{P(x)}.dx = \int P(x).log\frac{P(x)}{Q(x)}.dx, & \textit{for continuous distributions} \end{cases}$$

These are the different loss functions of regressions.