Loss Functions

Regression Losses

- **Mean Square Error (or) Quadratic Loss (or) L2 Loss**
- **Mean Absolute Error (or) L1 Loss**
- **Mean Bias Error**

Classification Losses

- Hinge Loss
- Cross-entropy loss (or) log loss
- Likelihood Loss

Loss functions

- Loss functions are a type of methods to evaluate how well your algorithm models your dataset. If your predictions are totally off, your loss function will output a higher number. If they're pretty good, it'll output a lower number. As you change pieces of your algorithm to try and improve your model, your loss function will tell you if you're getting anywhere.
- Gradually, with the help of some optimization function like Gradient Descent, loss function learns to reduce the error in prediction.
- There's no one-size-fits-all loss function to algorithms in machine learning. There are various factors involved in choosing a loss function for specific problem such as type of machine learning algorithm chosen, ease of calculating the derivatives and to some degree the percentage of outliers in the data set.

Loss functions

- We can classify loss functions into two major categories depending upon the type of learning task we are dealing with like Regression losses or Classification losses.
- In classification, we are trying to predict the output from a set of finite categorical values.
- In Regression, on the other hand, deals with predicting a continuous value.

Regression losses

- MSE / Quadratic Loss/ L2 Loss
- MAE / L1 loss
- Mean bias error

Classification losses

- Hinge loss
- Cross - entropy loss (or) log loss
- Likelihood loss

We will discuss every loss function in detail in the next Posts.

Loss functions

MSE / Quadratic Loss/ L2 Loss

- Mean Squared Error, or MSE loss is the default loss to use for regression problems.
- Mathematically, it is the preferred loss function under the inference framework of maximum likelihood if the distribution of the target variable is Gaussian. It is the loss function to be evaluated first and only changed if you have a good reason.

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

- y is the actual values.
- y hat is the predicted values.
- n is the total number of samples.Add subheading

- Mean squared error is calculated as the average of the squared differences between the predicted and actual values.
- The result is always positive regardless of the sign of the predicted and actual values and a perfect value is 0.0.
- The squaring means that larger mistakes result in more error than smaller mistakes, meaning that the model is punished for making larger mistakes.

Loss functions

Mean Absolute Error / L1 Loss

- On some regression problems, the distribution of the target variable may be mostly Gaussian but may have outliers, e.g. large or small values far from the mean value.

$$MAE = \frac{\sum_{i=1}^{n} | y_i - \hat{y}_i |}{n}$$

  - y is the actual values.
  - y hat is the predicted values.
  - n is the total number of samples.Add subheading

- Mean absolute error, on the other hand, is measured as the average sum of absolute differences between predictions and actual observations.
- Like MSE, this as well measures the magnitude of error without considering their direction.
- Unlike MSE, MAE needs more complicated tools such as linear programming to compute the gradients.
- Plus MAE is more robust to outliers since it does not make use of square.

Loss functions

Mean Bias Error

- This is much less common in machine learning domain as compared to its counterpart.
- This is the same as MSE with the only difference that we don't take absolute values.
- Clearly, there's a need for caution as positive and negative errors could cancel each other out. Although less accurate in practice, it could determine if the model has a positive bias or negative bias.

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

- y is the actual values.
- y hat is the predicted values.
- n is the total number of samples.Add subheading

Loss functions

Cross-Entropy Loss (Binary Classification)

- Cross-entropy is the default loss function to use for binary classification problems.

$$CrossEntropyLoss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i))$$

  - y is the actual values.
  - y hat is the predicted values.

- It is intended for use with binary classification where the target values are in the set {0, 1}.
- Mathematically, it is the preferred loss function under the inference framework of maximum likelihood. It is the loss function to be evaluated first and only changed if you have a good reason.
- Notice that when actual label is 1 (y(i) = 1), second half of function disappears whereas in case actual label is 0 (y(i) = 0) first half is dropped off.
- In short, we are just multiplying the log of the actual predicted probability for the ground truth class. An important aspect of this is that cross entropy loss penalizes heavily the predictions that are confident but wrong.

Loss functions

Hinge Loss (Binary Classification)

- An alternative to cross-entropy for binary classification problems is the hinge loss function, primarily developed for use with Support Vector Machine (SVM) models.
- It is intended for use with binary classification where the target values are in the set {-1, 1}.
- The hinge loss function encourages examples to have the correct sign, assigning more error when there is a difference in the sign between the actual and predicted class values.
- Reports of performance with the hinge loss are mixed, sometimes resulting in better performance than cross-entropy on binary classification problems.

We learn more about this when we learn SVM algorithm