https://cestlaviewellness.ca/wp-content/uploads/2021/01/Loss-blog.jpg

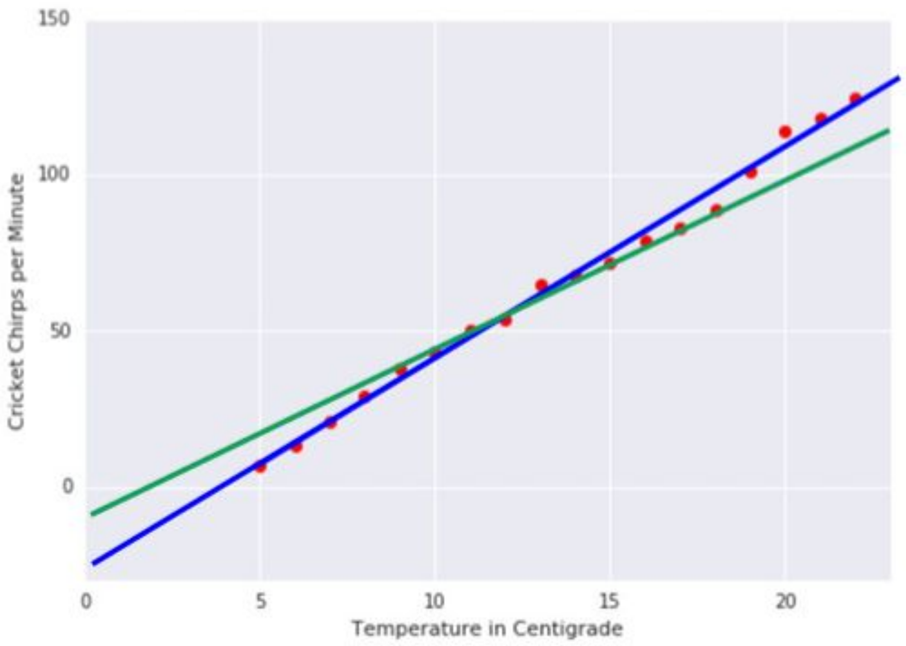
Loss Functions

PhD. Msc. David C. Baldears S.
PhD(s). Msc. Diego Lopez Bernal

TC3007C

Loss

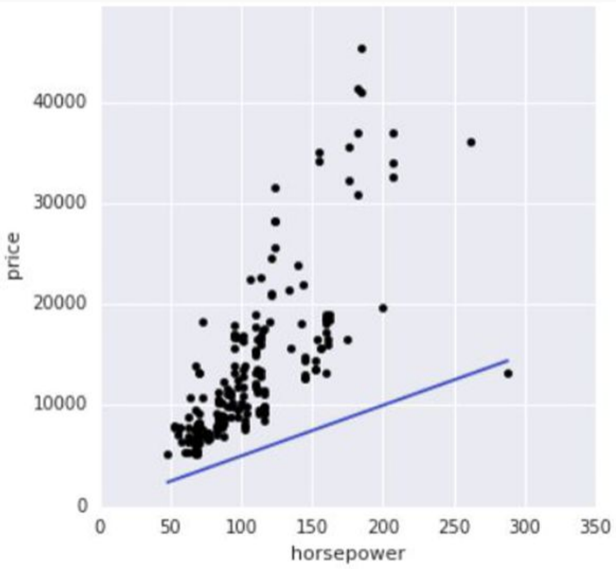
- There must be a way to say what solution fits the data best.
- Loss is a penalty for an incorrect prediction.
- Lower loss is generally better.
- Choice of loss function guides the model chosen.

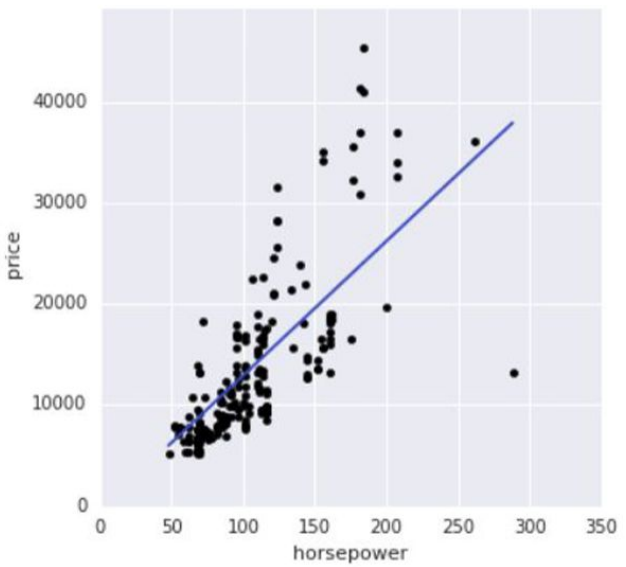


- Do you think the green line or blue line will lead to better predictions?

Evaluating If a Model Is Good

- It is common to look at mathematical ways to evaluate the quality of a model but that should not replace more informal yet intuitive ways to evaluate a model.
- When you just have one variable, drawing a line on a scatter plot that shows the model predictions is a good tool.
- Let's look at two different models to predict the price of a car from its engine's horsepower.
- Which Model is Better?





Measuring Loss

- What are some mathematical ways to measure loss?
  - Remember that
  - Loss should become smaller when predictions improve.
  - Loss should become larger when prediction get worse.
- Lots of ideas are reasonable here; be creative.

Mean Absolute Error

- Mean Absolute Error(MAE), or L1 loss.
- L1 Loss function minimizes the absolute differences between the estimated values and the existing target values. So, summing up each target value y_i and corresponding estimated value $h(x_i)$, where x_i denotes the feature set of a single sample.
- Sum of absolute differences for ‘n’ samples can be calculated as

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|h(x_i) - y_i\right|$$

MAE - Mean Absolute Error
- n - number of samples
- x_i - i-th sample from dataset
- $h(x_i)$- prediction for i-th sample (thesis)
- y_i - ground truth label for i-th sample

Mean Square Error

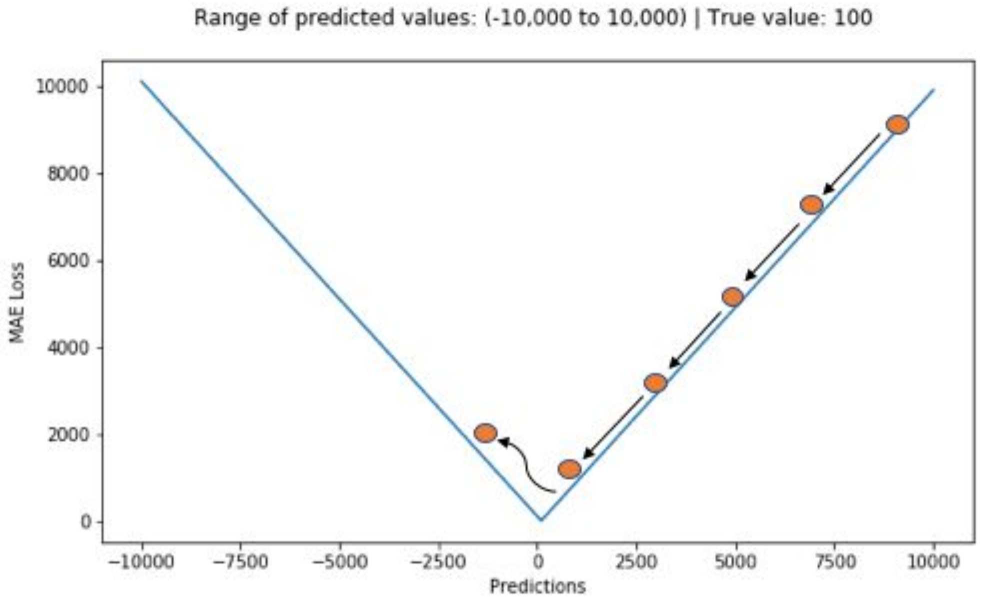
- Mean Squared Error(MSE), or L2 loss.
- L2 loss function minimizes the squared differences between the estimated and existing target values.
- = square of the difference between prediction and label
- = $(y - y')^2$

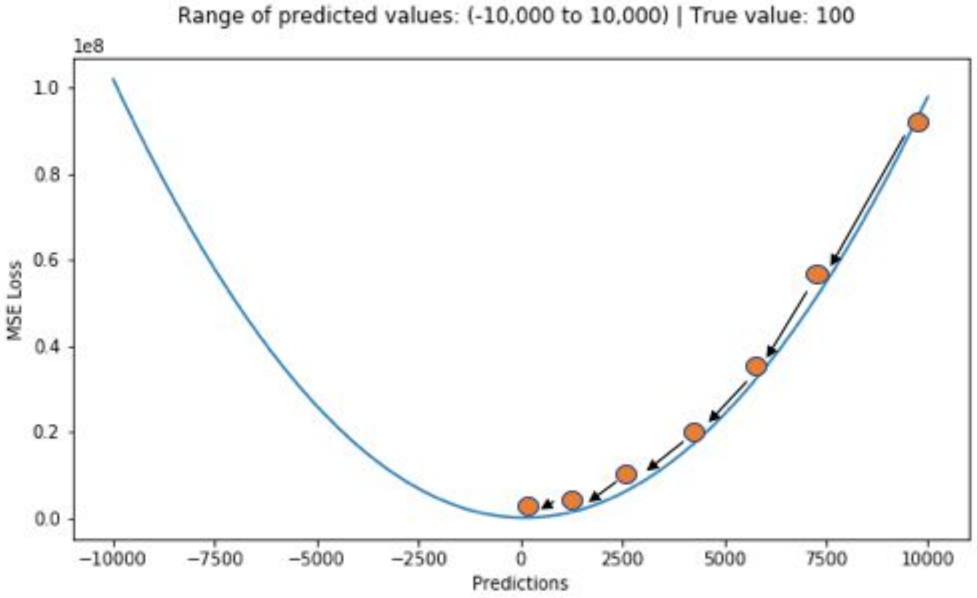
$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(h(x_i) - y_i\right)^2$$

MSE - mean square error

- n - number of samples
- x_i - i-th sample from dataset
- $h(x_i)$- prediction for i-th sample (thesis)
- y_i - ground truth label for i-th sample

MAE vs MSE


Range of predicted values: (-10,000 to 10,000) | True value: 100
MAE Loss
Predictions
10000
8000
6000
4000
2000
0
−10000
−7500
−5000
−2500
0
2500
5000
7500
10000


Range of predicted values: (-10,000 to 10,000) | True value: 100
1e8
MSE Loss
Predictions
1.0
0.8
0.6
0.4
0.2
0.0
−10000
−7500
−5000
−2500
0
2500
5000
7500
10000

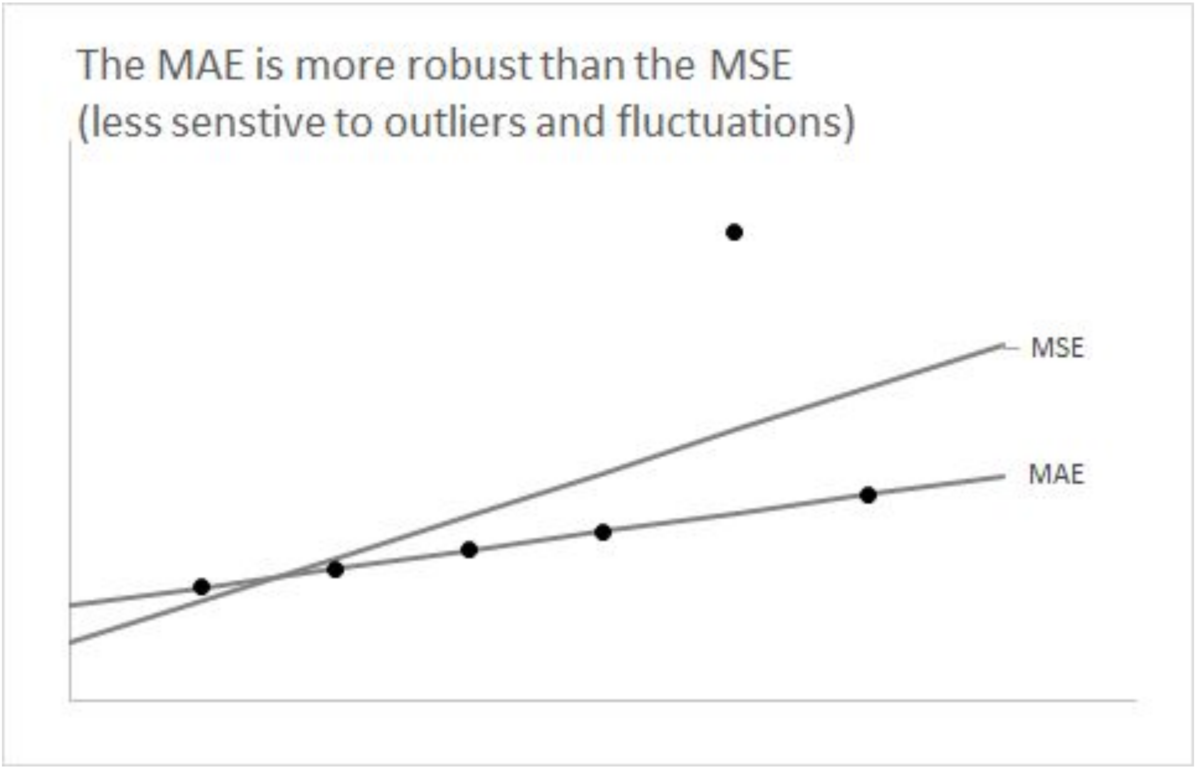
Differences between L1 and L2

The differences of L1-norm and L2-norm as a loss function can be promptly summarized as follows:

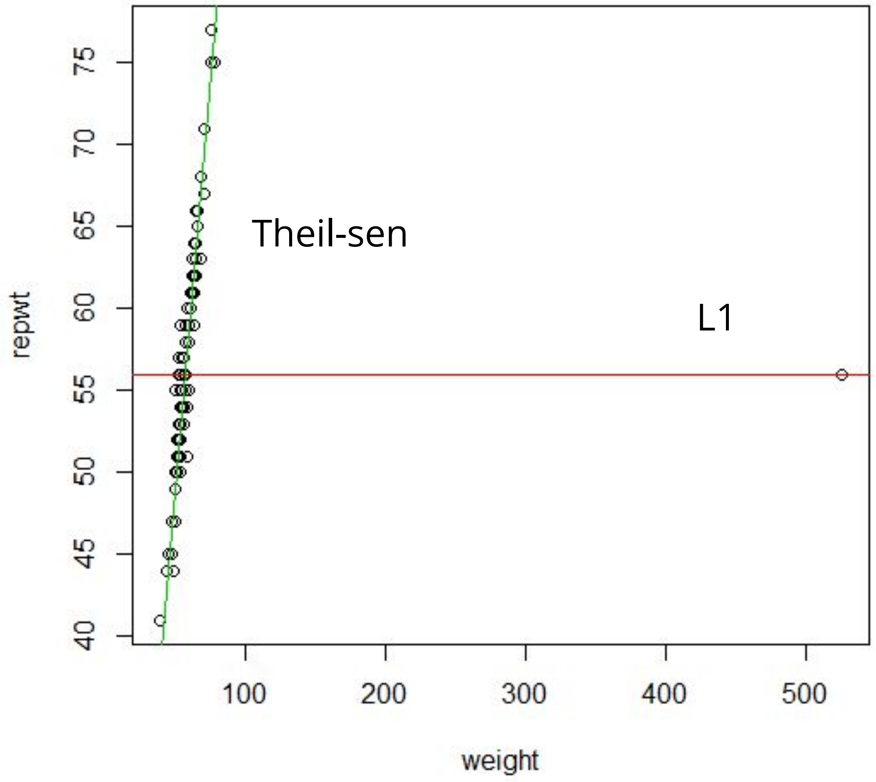
L2 loss function	L1 loss function
Not very robust	Robust
Stable solution	Unstable solution
Always one solution	Possibly multiple solutions

Differences between L1 and L2

Robustness: how affected they are by outliers.

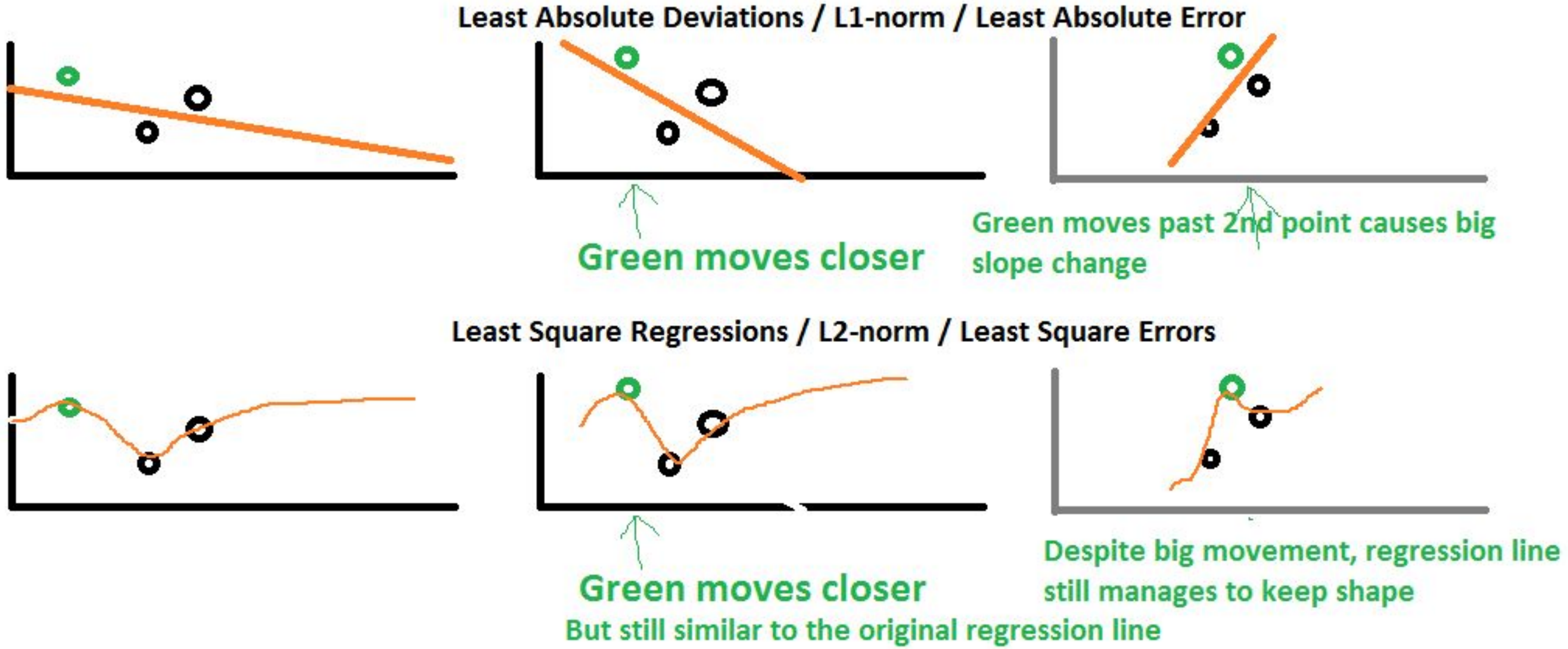


Note: Sometimes L1 is also not so robust.



Differences between L1 and L2

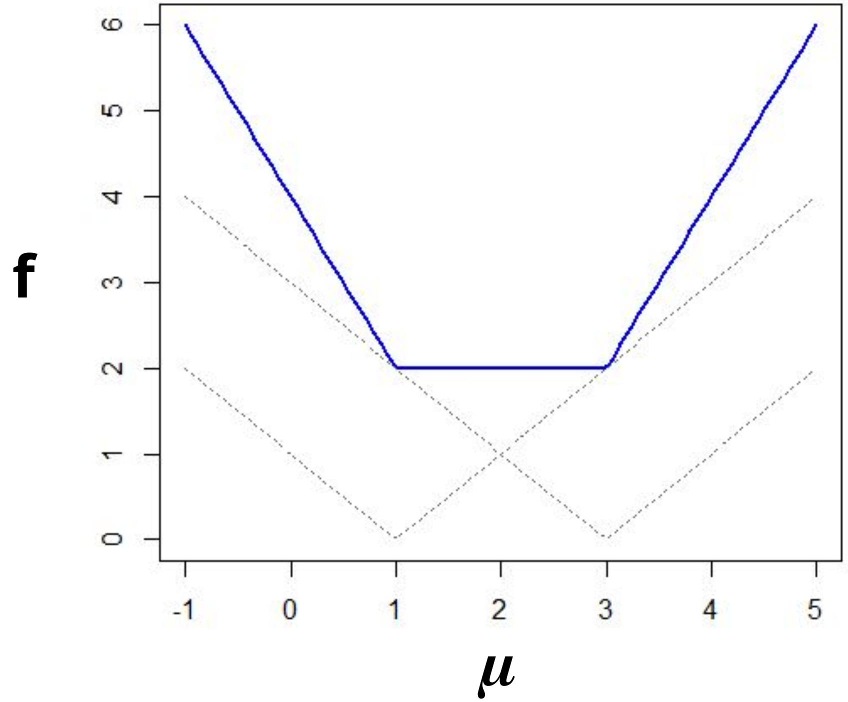
Stability: how much the solution changes if a data point changes

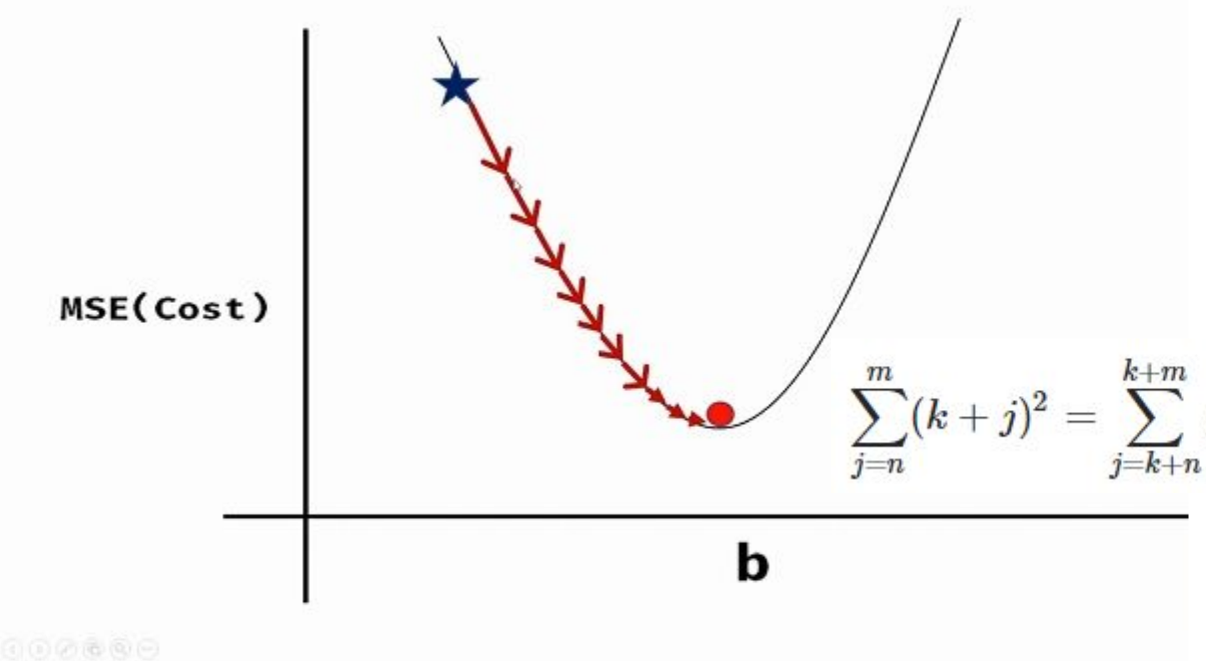


Differences between L1 and L2

Amount of solutions: while both $|x-\mu|$ and $(x-\mu)^2$ have a single minimum, the sum of absolute values often does not. Consider $x_1=1$ $x_2=3$

Note: Sum of quadratics still quadratics





$$\sum_{j=n}^{m}(k+j)^2 = \sum_{j=k+n}^{k+m} j^2 = \sum_{j=1}^{k+m} j^2 - \sum_{j=1}^{k+n-1} j^2$$

Huber

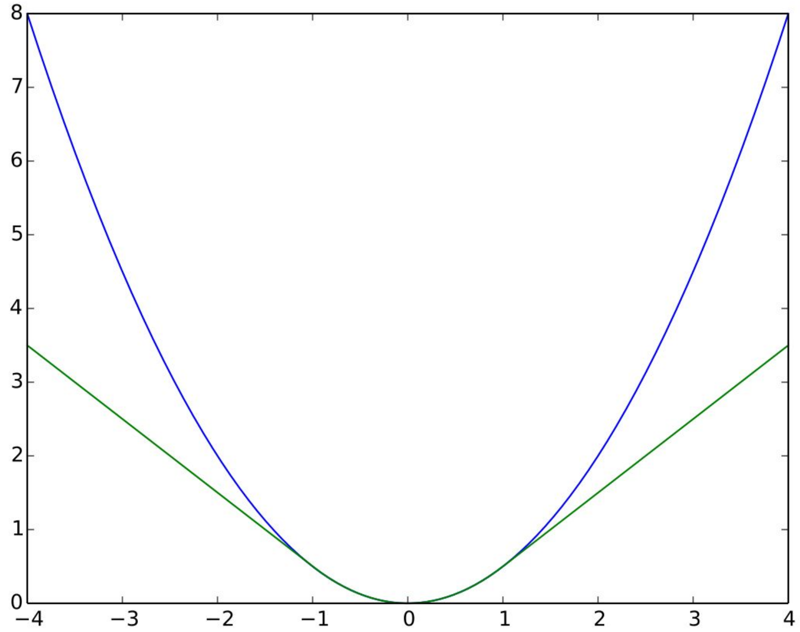
Typically used for regression. It's less sensitive to outliers than the MSE as it treats error as square only inside an interval.

Quadratic

$$L_\delta = \begin{cases} \frac{1}{2}\left(y - f(x)\right)^2, & if\ |y - f(x)| \le \delta \\ \delta|y - f(x)| - \frac{1}{2}\delta^2, & otherwise \end{cases}$$

Linear

The squared loss has the disadvantage that it has the tendency to be dominated by outliers



Huber loss (green, $\delta = 1$) and squared error loss (blue) as a function of $y - f(x)$

https://en.wikipedia.org/wiki/Huber_loss

Root Mean Square Error

Root Mean Square Error (RMSE) is a frequently used measure of the differences between values (sample or population values) predicted by a model or an estimator and the values observed.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(h(x_i) - y_i\right)^2}$$

RMSE - root mean square error

- n - number of samples
- x_i - i-th sample from dataset
- $h(x_i)$- prediction for i-th sample (thesis)
- y_i - ground truth label for i-th sample

Entropy

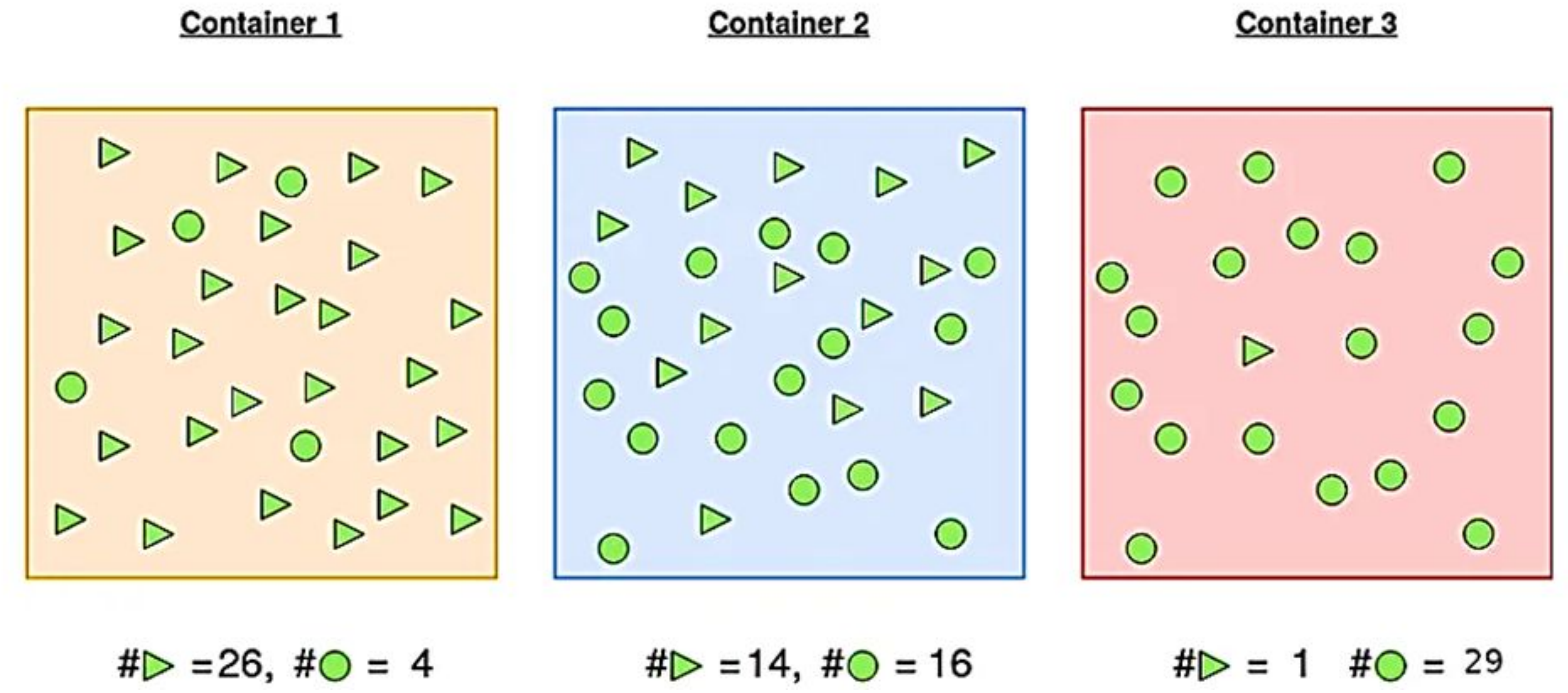
- Entropy of a random variable X is the level of uncertainty inherent in the variables possible outcome. Greater H(X), greater uncertainty.

$$H(X) = \begin{cases} -\int_x p(x)\log p(x), & \text{if } X \text{ is continous} \\ -\sum_x p(x)\log p(x), & \text{if } X \text{ is discrete} \end{cases}$$

P(x) -> probability of x

Entropy

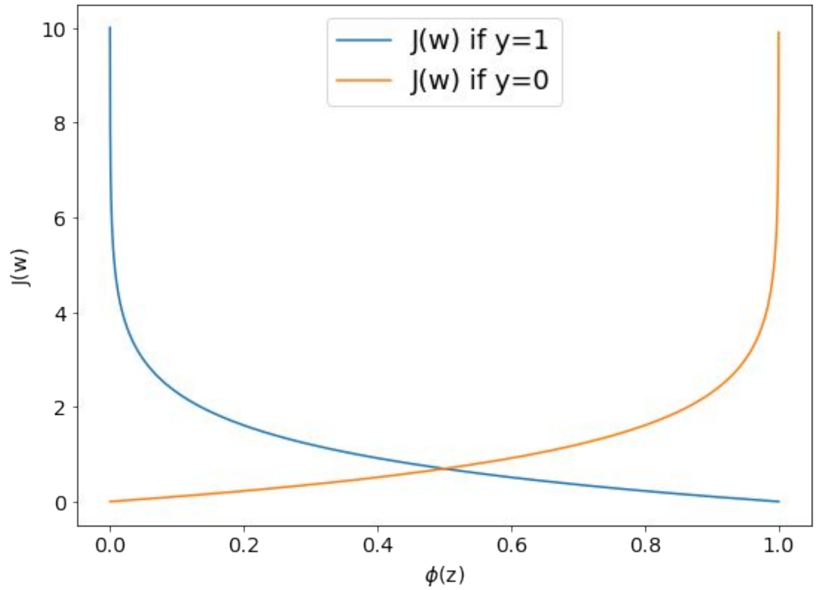
Example: calculate the entropy to know the uncertainty on choosing a given shape from a container.



Cross-Entropy loss

- Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1.
- Cross-entropy loss increases as the predicted probability diverges from the actual label.
  - So predicting a probability of .012 for label 1 when the actual observation label is 1 would be bad and result in a high loss value.
  - A perfect model would have a log loss of 0.
- It is defined as:

$$L_{CE} = -\sum_{i=1}^{n} y_i log(\pi_i)$$



Explanation of Cases of Cost Function

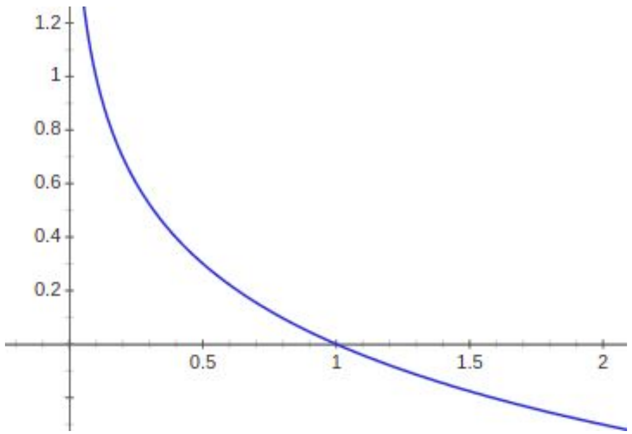
$$J(\beta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i ln[\pi(x_i) + (1-y_i) ln[1-\pi(x_i)]\right]$$

When Y = 1:

Penalize as it moves farther from 1,
-log(π(x))

h(x)	-log(h(x))	-log(1-h(x))
0	inf	0
0.1	2.30258509	0.10536052
0.2	1.60943791	0.22314355
0.3	1.2039728	0.35667494
0.4	0.91629073	0.51082562
0.5	0.69314718	0.69314718
0.6	0.51082562	0.91629073
0.7	0.35667494	1.2039728
0.8	0.22314355	1.60943791
0.9	0.10536052	2.30258509
1	0	inf

Cost

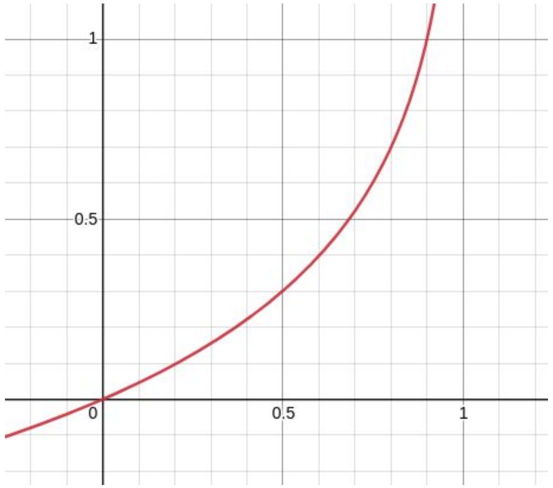


Model Prediction

When Y = 0:

Penalize as it moves farther from 0,
-log(1- π(x))

Cost



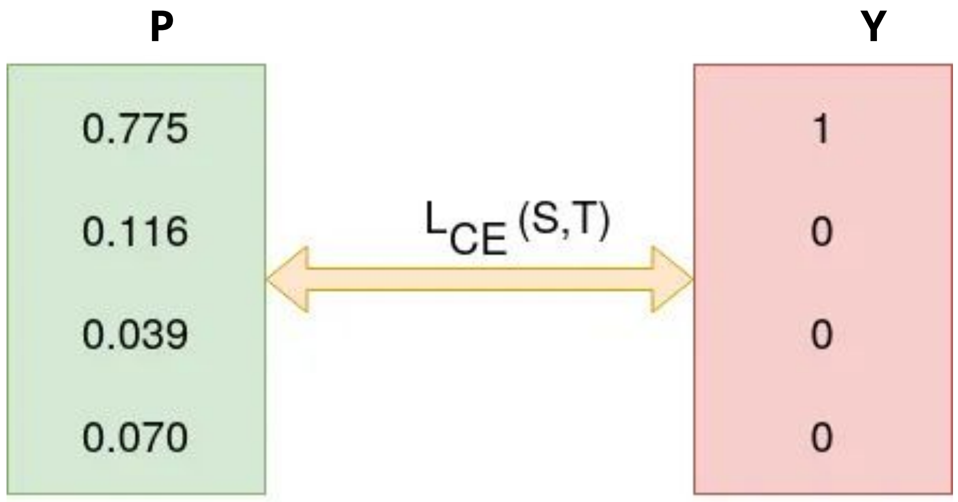
Model Prediction

Note: our model's prediction won't exceed 1 and won't go below 0. So, that part is outside of our worries.

$$J(\beta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i ln[\pi(x_i) + (1-y_i) ln[1-\pi(x_i)]\right]$$

Example of Binary Cross Entropy

Calculate the cross entropy loss of the following classifier:



$$L_{CE1} = -[1 * log(0.775) + (1-1) * log(1-0.775)] = 0.25489$$

$$L_{CE2} = -[0 * log(0.116) + (1-0) * log(1-.116)] = 0.1282$$

$$L_{CE3} = -[0 * log(0.039) + (1-0) * log(1-.039)] = 0.0397$$

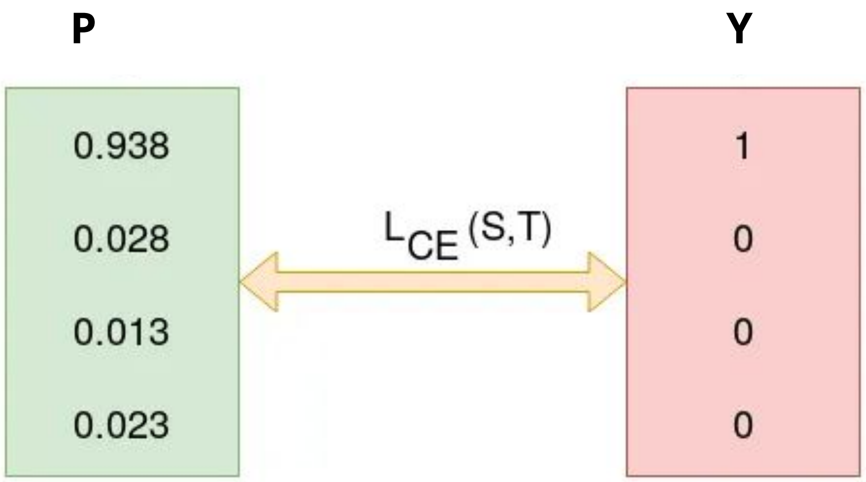
$$L_{CE4} = -[0 * log(0.070) + (1-0) * log(1-.070)] = 0.0725$$

$$Avgloss = 0.1238$$

$$J(\beta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i ln[\pi(x_i) + (1-y_i) ln[1-\pi(x_i)]\right]$$

Example of Binary Cross Entropy

Calculate the cross entropy loss of the following classifier:



What do you expect? Higher or lower loss?

Cross-Entropy loss

- **Cross-Entropy = 0.00**: Perfect probabilities.
- **Cross-Entropy < 0.02**: Great probabilities.
- **Cross-Entropy < 0.05**: On the right track.
- **Cross-Entropy < 0.20**: Fine.
- **Cross-Entropy > 0.30**: Not great.
- **Cross-Entropy > 1.00**: Terrible.
- **Cross-Entropy > 2.00** Something is broken.

Multiclass Cross Entropy

In binary classification, where the number of classes M equals 2, cross-entropy can be calculated as:

$$-(ylog(p) + (1 - y)log(1 - p))$$

If M>2 (i.e. multiclass classification), we calculate a separate loss for each class label per observation and sum the result.

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c})$$

- M - number of classes (dog, cat, fish)
- log - the natural log, aka ’ln’
- y - binary indicator (0 or 1) if class label **c** is the correct classification for observation **o**
- p - predicted probability observation **o** is of class **c**

Multiclass Cross Entropy