COMP2261 ARTIFICIAL INTELLIGENCE / MACHINE LEARNING

Cost Functions

For Regression Models

-- Log-Cosh

Dr SHI Lei

Durham University

learning lab

Learning Objectives

- Understand what is Log-Cosh cost
- Understand how Log-Cosh cost function works
- Understand why use Log-Cosh cost

MSE

Quantile

MAE

Log-Cosh

Huber-M

Log-Cosh

Log-Cosh

$$J = \frac{1}{m}\sum_{i=1}^{m} \log(\cosh(y^{(i)} - \hat{y}^{(i)}))$$

Log-Cosh

$$J = \frac{1}{m}\sum_{i=1}^{m} \boxed{\log(\cosh(y^{(i)} - \hat{y}^{(i)}))}$$

$$\mathrm{f(x)} = \log(\cosh(x)) = logcosh(x) = ln\frac{e^x + e^{-x}}{2}$$

If x is very large

$$x > 0 \qquad = ln\frac{e^x + \cancel{e^{-x}}}{2} = ln\frac{e^x}{2} = ln(e^x) - ln2 = x - ln2$$

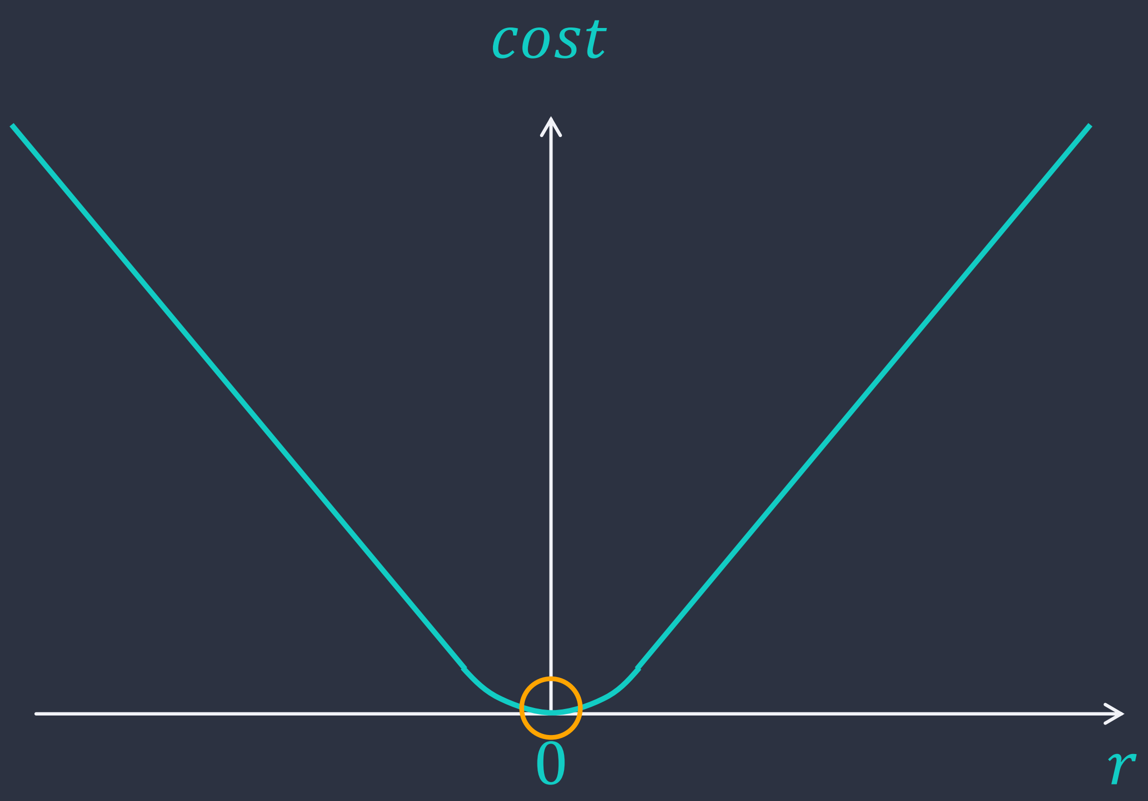
$$x < 0 \qquad = ln\frac{\cancel{e^x} + e^{-x}}{2} = ln\frac{e^{-x}}{2} = ln(e^x) - ln2 = -x - ln2$$

If x is very large $\quad \log(\cosh(x)) = |x| - ln2$

If x is very small $\quad \log(\cosh(x)) = \frac{x^2}{2} - \cancel{\frac{x^4}{12}} + \cancel{\frac{x^6}{45}} - \cancel{\cdots} = \frac{x^2}{2}$

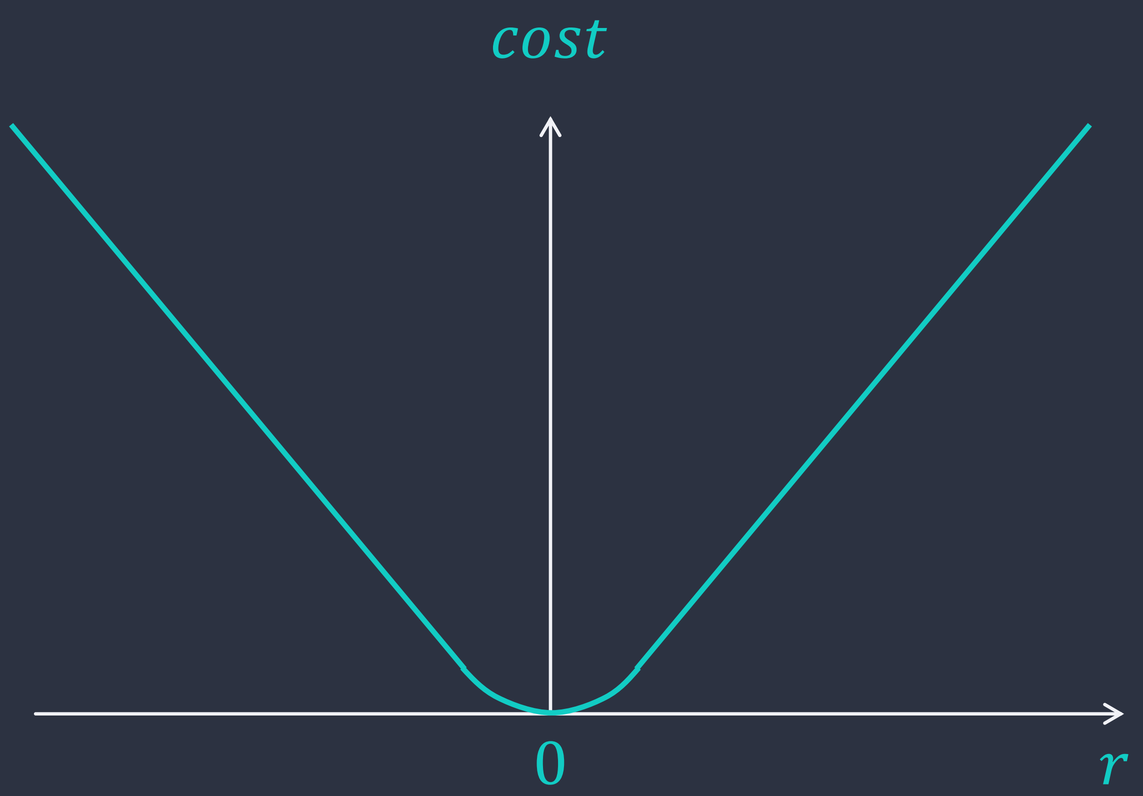
Log-Cosh

$$J = \frac{1}{m}\sum_{i=1}^{m} \log(\cosh(y^{(i)} - \hat{y}^{(i)}))$$



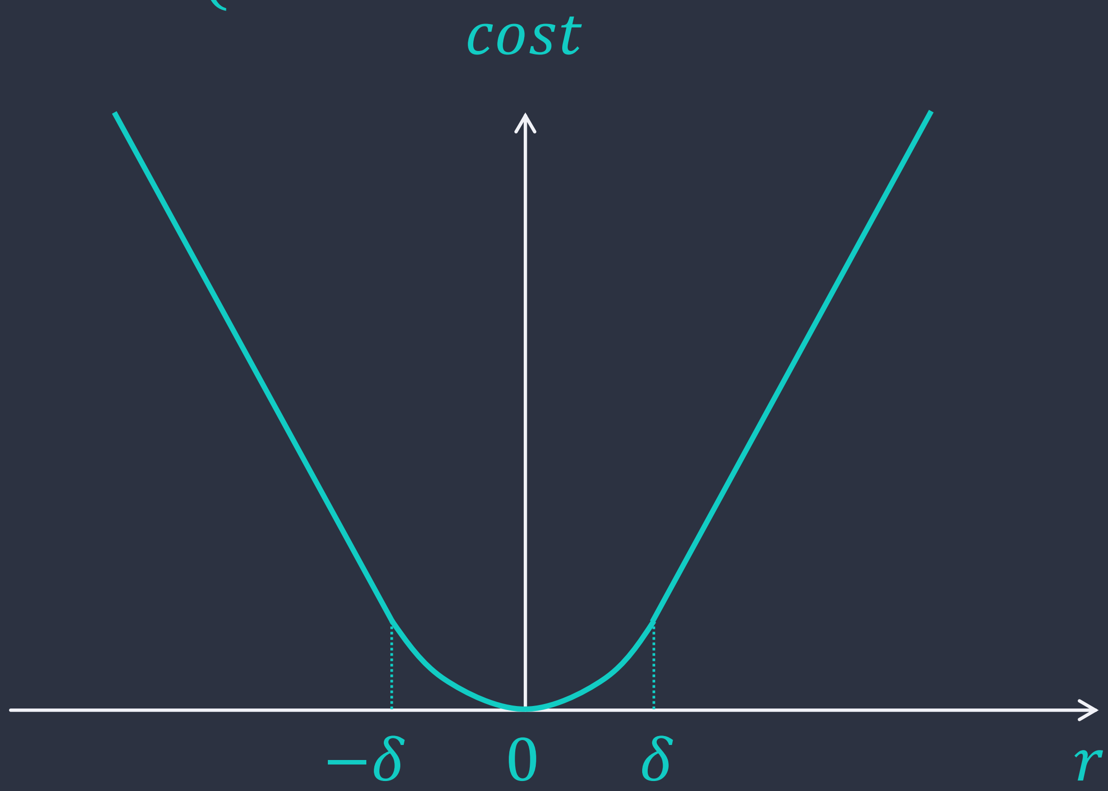
Log-Cosh

$$J = \frac{1}{m}\sum_{i=1}^{m} \log(\cosh(y^{(i)} - \hat{y}^{(i)}))$$



Huber-M

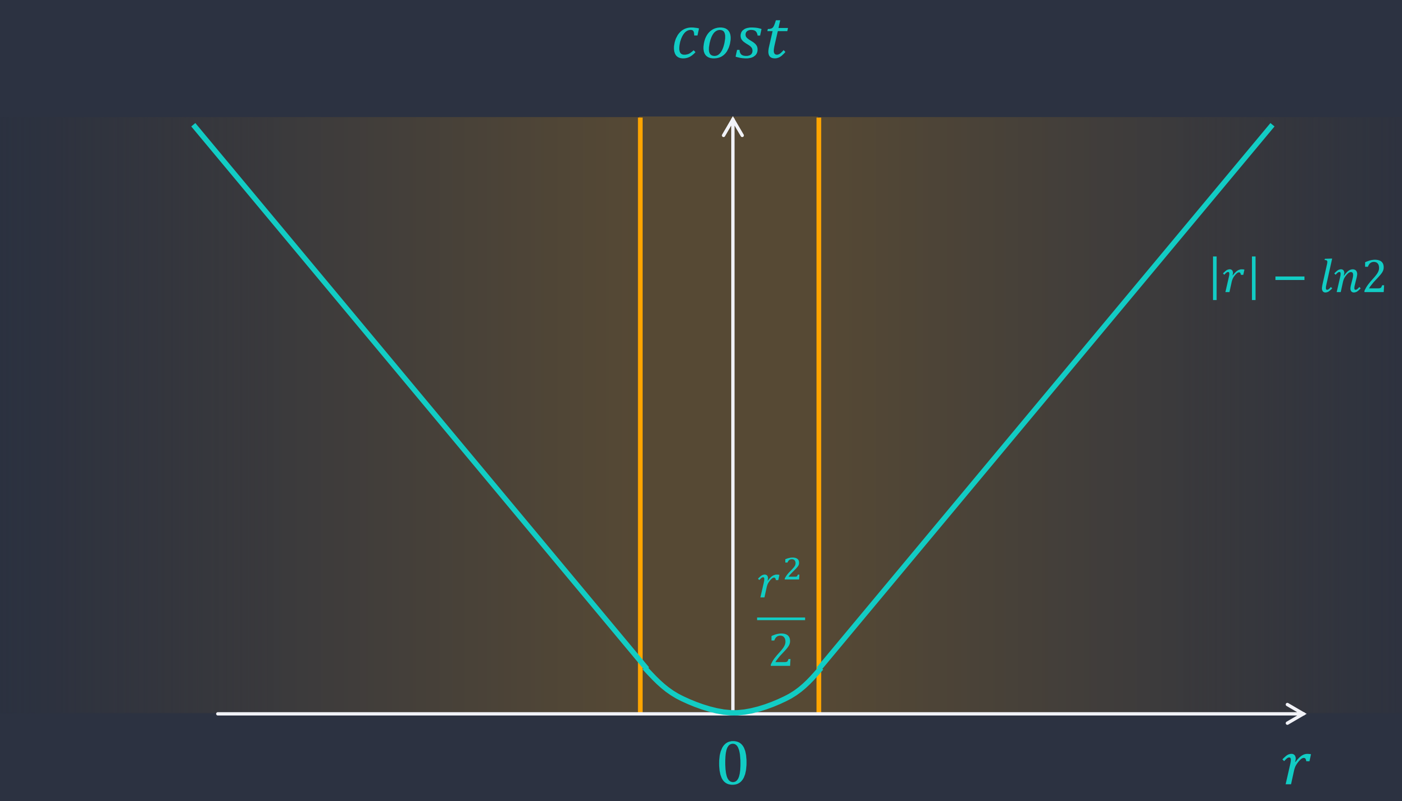
$$J = \frac{1}{m}\sum_{i=1}^{m} \begin{cases} \frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2, & \left|y^{(i)} - \hat{y}^{(i)}\right| \leq \delta \\ \delta\left(\left|y^{(i)} - \hat{y}^{(i)}\right| - \frac{1}{2}\delta\right), & otherwise \end{cases}$$



Log-Cosh

$$J = \frac{1}{m}\sum_{i=1}^{m} \log(\cosh(y^{(i)} - \hat{y}^{(i)}))$$

Twice differentiable everywhere.



Takeaway Points

- Log-Cosh cost has the good properties of Huber-M cost.
  - Differentiable everywhere -> good for gradient descent.
  - Less sensitive to outliers.
- No need hyperparameter tuning to decide a different calculation for different residuals (Huber-M needs tuning δ)
- Twice differentiable everywhere -> good for learning algorithms that need to use Newton's method to find the optimum.