# Logistic Evidence‑Weighting Rule – Quick Reference

## 1  What the Logistic Rule Does

A logistic conditional rule turns qualitative beliefs (“baseline plausibility” and “importance of each parent”) into a numeric probability that a binary hypothesis H is true. It is compact, interpretable, and smoothly bounded between 0 % and 100 %.

## 2  Inputs You Ask the User For

• \*\*Baseline probability p₀\*\* – Probability H would be true if none of its parents were true.

• \*\*Importance weight wᵢ\*\* for each parent Xᵢ – Positive supports H, negative undermines H.

  \*(Alternative wording\*: ask for an \*\*odds multiplier rᵢ\*\*; set wᵢ = ln rᵢ.)

## 3  Build the Logistic Equation

For k binary parents X₁ … X\_k (0 = false, 1 = true):

 β₀ = logit(p₀) = ln[p₀ / (1 − p₀)]

 βᵢ = wᵢ (or βᵢ = ln rᵢ if odds multiplier)

Probability that H is \*\*true\*\* given a parent configuration is then:

 P(H = True | X₁…X\_k) = σ(β₀ + β₁X₁ + … + β\_kX\_k)

where σ(z) = 1 / (1 + e^(−z)) is the logistic (sigmoid) function.

Because H is binary, P(H = False | …) = 1 − P(H = True | …).

## 4  Rationale for Using the Logistic Link

• \*\*Compact\*\* – Needs only k + 1 parameters instead of 2ᵏ rows in a full CPT.

• \*\*Additive in log‑odds\*\* – Each weight acts as an independent, intuitive “vote”.

• \*\*Smooth & bounded\*\* – Always yields a value between 0 % and 100 %.

• \*\*Learnable\*\* – Standard logistic regression/Bayesian GLMs can estimate the same parameters from data.

• \*\*Expressive\*\* – Positive, negative, or zero weights let you encode support, contradiction, or irrelevance without switching templates (noisy‑OR, noisy‑AND, etc.).

## 5  Illustrative Numeric Example

Inputs provided by user:

 • p₀ = 0.05 (baseline)

 • w₁ = 1.4 (parent X₁)

 • w₂ = 0.7 (parent X₂)

If X₁ = 1 and X₂ = 0:

 β₀ = ln(0.05 / 0.95) ≈ −2.94

 z = β₀ + 1.4·1 + 0.7·0 ≈ −1.54

 P(H = True) = σ(−1.54) ≈ 0.177 (17.7 %)

## 6  Implementation in Python (one‑liner)

import math  
def logistic(p0, betas, xs):  
 logit = lambda p: math.log(p/(1-p))  
 z = logit(p0) + sum(b\*x for b,x in zip(betas,xs))  
 return 1/(1+math.exp(-z))  
# logistic(0.05, [1.4,0.7], [1,0]) → 0.177