# Ethical Reasoning Tends to Associate Nearby Judgements With Usual Implied Meaning

by Sven Nilsen, 2018

Using the theory of ethics as rational reasoning with granular judgements about the world, I prove the following: When an ethical judgement `E(A)` is much closer to `E(A ∧ B)` than `E(A ∧ ¬B)` and the statements `A` and `B` are used in many different contexts, it is more likely that `A = B` than `A = ¬B`.

For example, the word “summer” in some countries is usually meant to imply “not cold”, because a summer that is not cold is judged to be closer to judgements about summers in general. This is true whether people think of cold summers as a good or bad thing.

The reason is that “summer” and “not cold” are used in many different contexts where the probability bias varies significantly. Overall, this means that the confidence factor falls within a normalized range (between 0 and 1) in more cases for closer judgements when the probability bias varies. Since the judgement is that “summers are not cold” is closer to the judgement of “summer” than the judgement of “summers are cold”, when somebody says “summer” they usually also mean “not cold”.

This was proven numerically using the following Dyon program:

```
fn main() {
    n := 10
    println(all i n+1, j n+1, k n+1 {
        e_a := i/n
        e_a_and_b := j/n
        e_a_and_not_b := k/n

        c_a_eq_b := \(p_b_given_a: f64) = 1 - (grab e_a - e_a_and_b) /
            (p_b_given_a * (grab e_a_and_b) + (1 - p_b_given_a) * (grab e_a_and_not_b) - grab e_a_and_b)
        c_a_eq_not_b := \(p_b_given_a: f64) = 1 - (grab e_a - e_a_and_not_b) /
            (p_b_given_a * (grab e_a_and_b) + (1 - p_b_given_a) * (grab e_a_and_not_b) - grab e_a_and_not_b)

        sum_prob_c_a_eq_b := sum i n+1 {if prob(\c_a_eq_b(i/n)) {1} else {0}}
        sum_prob_c_a_eq_not_b := sum i n+1 {if prob(\c_a_eq_not_b(i/n)) {1} else {0}}
        if sum_prob_c_a_eq_b > sum_prob_c_a_eq_not_b {
            abs(e_a - e_a_and_b) < abs(e_a - e_a_and_not_b)
        } else if sum_prob_c_a_eq_b < sum_prob_c_a_eq_not_b {
            abs(e_a - e_a_and_b) > abs(e_a - e_a_and_not_b)
        } else {
            true
        }
    })
}

prob(x: f64) = (x >= 0) && (x <= 1)
```