**Algorithm README**

Below is the mathematical explanation for the algorithm employed by the S.M.I.L.I.E app to give users feedback on the metrics they have been using to rate their day. Initially, all metrics are set to the same weight *DECIDE ON INITIAL WEIGHT* *CURRENTLY THINKING 2.0 – 2.5*. In order to work the algorithm needs multiple days worth of data *DECIDE ON INITIAL AMOUNT OF DATA NECESSARY FOR REFLECTION CURRENTLY THINKING 3 DAYS.* The following is how the calculation for the weights is performed:

**Step 1: Determine whether the user values the highly rated metrics more or the lower rated metrics more.**

The first and most intuitive step of the algorithm, we first determine the average of the actual ratings for the metrics:

Let mi represent the rating for metric i, and Ovr represent the overall rating given by the user for the day.

AvgRating = / *Number of Metrics*

Case 1: If AvgRating < Ovr, this indicates that the user puts more weight on the metrics that are higher than the AvgRating, proportionally

Case 2: If AvgRating > Ovr, this indicates that the user puts more weight on the metrics that are higher than the AvgRating, proportionally

Case 3: If AvgRating = Ovr, this indicates that the weights associated with each metric are proportional to the ratings given to them. In this case the proportional weight for each metric is easy to calculate. Separate the metric ratings into two sets, those that are less than the AvgRating and those that are more than the AvgRating. The weighting for each of the metric sets, is the inverse of the ratio of the size of one set to the other. For example: given ratings of 4, 5, 9, and an Ovr of 6. We would separate those ratings into sets of {4, 5} and {9}. Let L1 = {4, 5} and L2 = {9}. |L1| = 2, and |L2| = 1, so the weight of L1 is 1/ (2/1) = ½ and the weight of L2 = 1/(1/2) = 2. Then the true weight for each metric, mi is the product of the amount of influence mi has within each set multiplied by the weight of the set. So for example, them metric with the rating 5 would have a true weight of (5/(4+5)) \* (1/2) = 5/18

**Step 2: Determining the true weight coefficient for cases 1 and 2**

Let L1 represent the set of metrics that we have determined to be more important based on conditions in cases 1 and 2.

Let L2 represent the set of metrics that we have determined to be less important based on conditions in cases 1 and 2.

In order to determine appropriate weighting for each metric, we create a utility function using a form of utility theory.

In utility theory, in order to create a function that can show the utility for each possible outcome in a situation, you set the utility of the worst possible outcome to 0 and the utility of the best possible outcome to 1. You then calculate the utility of every value in between the best and worst by multiplying the best and worst outcomes numerical values by their utility values, adding them together, and dividing by the numerical value of each outcome. *INSERT PICTURE FROM MSCI NOTES HERE*

We proceed by doing something similar.

We assign the metric(s) from L1 with the smallest difference between their rating(s) and the Ovr a weight of 1. Let this metric be called Mv.

We assign the metric(s) from L2 with the largest difference between their rating(s) and the Ovr a weight of the reciprocal of the difference. Let this metric be called Mn.

For all remaining metrics, we calculate their weight the following way: Let mi represent any metric.

mi.rating \* mi.weight = Mv.rating \* Mv.weight + Mn.rating \* Mn.weight

if mi.weight > 1, mi.weight = 1/m­i.weight

Once we have done this for every metric, and we have a proportional weight, we now need to determine the true weight, i.e. the weight proportional to Ovr and not the AvgRating.

In order to this, we multiply each metric’s weight by the following coefficient:

Coef = (*Number of metrics \* Ovr) /*

And the new weight for each metric is mi.weight \* Coef

**Step 3: Adding new weight coefficient to previous weight data to get weight for metric**

In order to account for peaks and troughs and to keep the data accurate, we will run weights through the following function: *MAYBE CHANGE THE FUNCTION IF WE WANT SMALLER CHANGES IN WEIGHTS*

The weights associated with L1 will be inputted as positive values and the weights associated with L2 will be inputted as negative values, and the weight put through the function will be added to the existing weight for the metrics.

Note: If either one of L1 or L2 is empty, the input should be considered invalid and the user should re-input their data.

For example: if the user has ratings of 9, 8, 7, but an overall rating of 3, there is clearly a missing metric, and the user should be prompted to add it.

**Step 4: Smoothing (MAYBE)**

Once all of the metric weights have been updated, they should be resized to the initial sum of all weights. For example, if there were 11 metrics that all started at 2, and now they sum to x, they should be multiplied by 22/x

Once the weight for those metrics has reached *SOME VALUE (PERHAPS DOUBLE THE ORIGINAL WEIGHT OR SOME AMOUNT LARGER THAN THE AVG),* we notify the user that this is either an important metric and that they should focus on it and improve it, or that the metric is unimportant and should be removed.