La funzione di acquisto era inizialmente così:

if ((i-stock / i-stock-threshold ) <= (utility-of-best-product )\*trigger)

con stock-threshold= 1

con trigger= 1

Ragionandoci siamo giunte alla conclusione che:

* scritta così stiamo confrontando 2 quantità non confrontabili
* ragionando su excel con varie casistiche, risultava che gli utenti a volte comprassero nei momenti sbagliati e non si riesce a trovare un valore di trigger tale da far funzionare l’equazione

A screenshot of a graph

Description automatically generated

Successivamente abbiamo aggiunto il budget (in quanto è realistico che un utente non abbia capacità infinita di acquisto) e un adjustment factor

* if (i-price <= tot-budget) [
* let adjustment-factor 0  
   if (i-stock / i-stock-threshold) <= 1 [  
   ;es 0.3  
   ;adj sarebbe 1.7  
   ; se fosse 1+ stock/trhs --> adj = 1.3  
   set adjustment-factor ( 1 + (1 - i-stock / i-stock-threshold) ) ; Amplifica proporzionalmente a quanto manca per raggiungere il threshold  
   ]
* set p-utility p-init-utility \* ( adjustment-factor )
* l’adjustment factor considerate inialmente era (1-stock-i/stock-threshold-i)
* anche questa formula è sbagliata in quanto porta gli utenti ad acquistare sempre

infine abbiamo unito le 2 modifiche e siamo giunte a

if ((1 - i-stock / i-stock-threshold ) \* trigger <= (utility-of-best-product )) and (i-price <= tot-budget) [

che però continua ad avere dei problemi

)

Poi anche qualità influenzerà anche il costo di produzione🡪 più il prodotto è sostenibile, più il costo di produzione si avvicinerà a baseline\*(1+delta)

Se sust=1 allora costo prod aumenta del 10%

Price-min= costo prod-baseline

Exp = esponente compreso tra 0 e 1

Iniziamo a settare exp = 0.5 e poi vediamo

Di conseguenza mettiamo vincolo su che non può essere <0

C-memory = sottolista 1 (c1-1 c2-1 c3-1..) ;; sottolista 2 (c1-2 c2-2;; sottolista 3…..

Weight associato alla sottolista

While k < c-memory -length (k<10)

(((