Value-sensitive product (content) recommendation

Aditya Kiran

July 29, 2021

1 Abstract

2 Introduction

3 Formulation

For each article A_i in our database we know its 3 properties:

1. v_i^1 **is a vector of its graph-representation**:
   Let (V^1, E^1) be the coviewership graph. Here V^1 represents the set of graph-embeddings of all nodes (articles) v^1, and E^1 represents set of all edges connecting those nodes. d_{ij}^1's be the distance between v_i^1 and v_j^1.

2. v_i^2 **is a vector of its sentence embedding**:
   Let (V^2, E^2) be the similarity graph. As above, d_{ij}^2's be the distance between v_i^2 and v_j^2.

3. c_i **is the value associated with that article**

So, each article A_i is represented by this triad $\{v_i^1, v_i^2, c_i\}$. Although, this is a very general formulation. For the sake of simplicity, for now we will just assume that each article is characterized by its sentence embedding and its value.

In this work, we will use Healthline content recommendation as a setting usecase to demonstrate our idea.

Problem Statement:
Now, the problem statement is:
Given that a user is reading an article A_i, what article must be recommended next so that we

1. maximize the content relevance to the user

2. maximize the ad-monetary value.

Mathematically, given A_i, we seek A_j such that:

$$j = \arg\max_J \phi(A_i, A_J),$$

where ϕ is the utility function. This above formulation is mainly for our *Read Next* pages, but it could be easily extended to *Infinite scroll*.

How do we choose the utility function ϕ?

Each article A_J is attributed with its sentence-embedding and its value. So we must design a utility function ϕ that maps its attributes to a real value:

$$\phi : \mathbb{R}^D \times \mathbb{R}^D \times \mathbb{R} \ \to \mathbb{R}$$

- ϕ must be directly proportional to the value of the article A_J being recommended

$$\phi(A_i, A_J) \propto c_J$$

- ϕ must be inversely proportional to the distance of the article A_J from the current article A_i

$$\phi(A_i, A_J) \propto \frac{1}{|d_i - d_J|} = \frac{1}{d_{iJ}}$$

Hence, we choose

$$\phi(A_p, A_q) := \frac{(c_q)^{r_1}}{(d_{pq})^{r_2}}.$$

Here r_1 and r_2 are the parameters that control the contribution of the value and the distance, on the utility term ϕ respectively.

Note: The special case of $\{r_1 = 0, r_2 = 1\}$ reduces the method back to the BAU. So once implemented, this method is easily backward compatible.

Assumptions made:

1. The value of an article $\approx$ its $\left(\frac{\text{total revenue}}{\text{total impressions}}\right)$ from the past
2. The article value remains almost unchanged for 1 week
3. Each article is characterized by the universal sentence embedding vector of its:
   - post_title
   - meta_title
   - summary

Formulation for Infinite scroll:

Quite often, we recommend more than just one article, either in the form of infinite-scroll or the '*Next Page*' option.

So a more general form of the problem is: Given an article A_i, we seek (N-1) articles $\{A_{a_1}, A_{a_2}, \cdots A_{a_{N-1}}\}$ such that:

$$\{a_1 \cdots a_{N-1}\} = \underset{\{b_1 \cdots b_{N-1}\}}{\arg\max} \sum_{k=1}^{N-1} \gamma^{k-1} \phi(A_i, A_{b_k}),$$

where $0 < \gamma < 1$ is called the discount factor.

Finally, after we know the 'optimal' recommendations, we evaluate the discounted value as

$$\mathcal{V}_{disc} := \sum_{k=1}^{N-1} \gamma^{k-1} c_{a_k}$$

We use the discounted-value instead of actual value to account for the fact that the likelihood of a user reading an article decreases with increasing scroll depth.

4 Results

Simulation results for *Infinite scroll*:

We used $r_1 = 0.8, r_2 = 15, \gamma = 0.09, N = 6$, and ran 30,000 random simulations.

- We observe a **lift of 14.76% in the discounted value**

$$\left(\frac{\mathcal{V}_{disc}^{VAL} - \mathcal{V}_{disc}^{BAU}}{\mathcal{V}_{disc}^{BAU}} \right) \times 100 = 14.76\%$$

  This lift can be interpreted as the revenue lift seen per impression on an average.

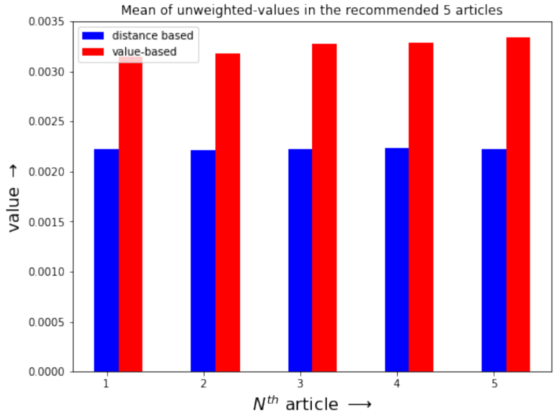
- We observe a **reduction in relevance or similarity of 0.4%**

  I.e., Mean cosine-similarity between the recommended articles and the article being read:

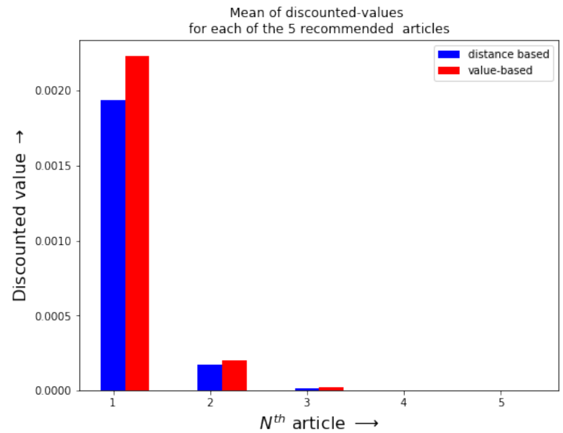
  - in BAU: 0.709
  - in VAL: 0.706.

```
Rand-expt #5/25000
Current article being read:-
  ['21 Chilled Soups That Are So Hot Right Now']

Recommended using distance-based:-
1. [disc_val=0.00135, cossim=0.7454] 9 Soups to Meal-Prep Now That It's Chilly Out and Salads Aren't Cutting It
2. [disc_val=0.00018, cossim=0.7386] 5-Ingredient Fall Soups That Don't Require Hours of Chopping
3. [disc_val=0.00002, cossim=0.7214] 19 Whole30 Soup Recipes That Make January Way More Tolerable
4. [disc_val=0.00000, cossim=0.7157] 19 Breakfast Soups (Yes, Really!) for Cold Mornings
5. [disc_val=0.00000, cossim=0.6856] 21 Noodle Soup Recipes That Prove It's Not All About Chicken

Recommended using value-based:-
1. [disc_val=0.00195, cossim=0.7386] 5-Ingredient Fall Soups That Don't Require Hours of Chopping
2. [disc_val=0.00012, cossim=0.7454] 9 Soups to Meal-Prep Now That It's Chilly Out and Salads Aren't Cutting It
3. [disc_val=0.00002, cossim=0.7214] 19 Whole30 Soup Recipes That Make January Way More Tolerable
4. [disc_val=0.00000, cossim=0.7157] 19 Breakfast Soups (Yes, Really!) for Cold Mornings
5. [disc_val=0.00000, cossim=0.6856] 21 Noodle Soup Recipes That Prove It's Not All About Chicken

 Value: Old= 0.00154     Value-new= 0.00209      lift= 35.68 %
Mean cosine_similarity: Old= 0.721       new= 0.721
```

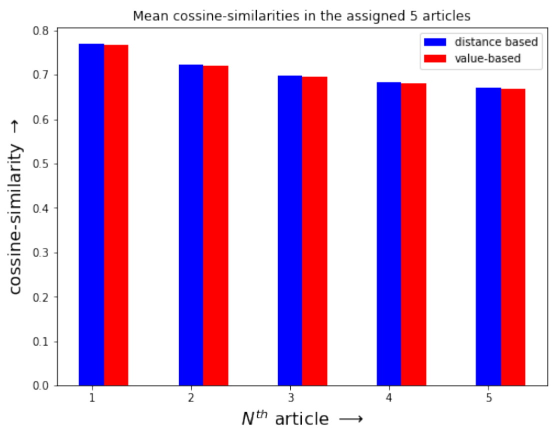
Figure 1: Example recommendations showing a lift in the value for the same set of articles.



(a) Mean of unweighted-values in the recommended $(N-1)$ articles



(b) Mean of the discounted-values



(c) Mean cosine-similarities

Figure 2: Conglomerated results of the recommended articles after doing the random simulations.

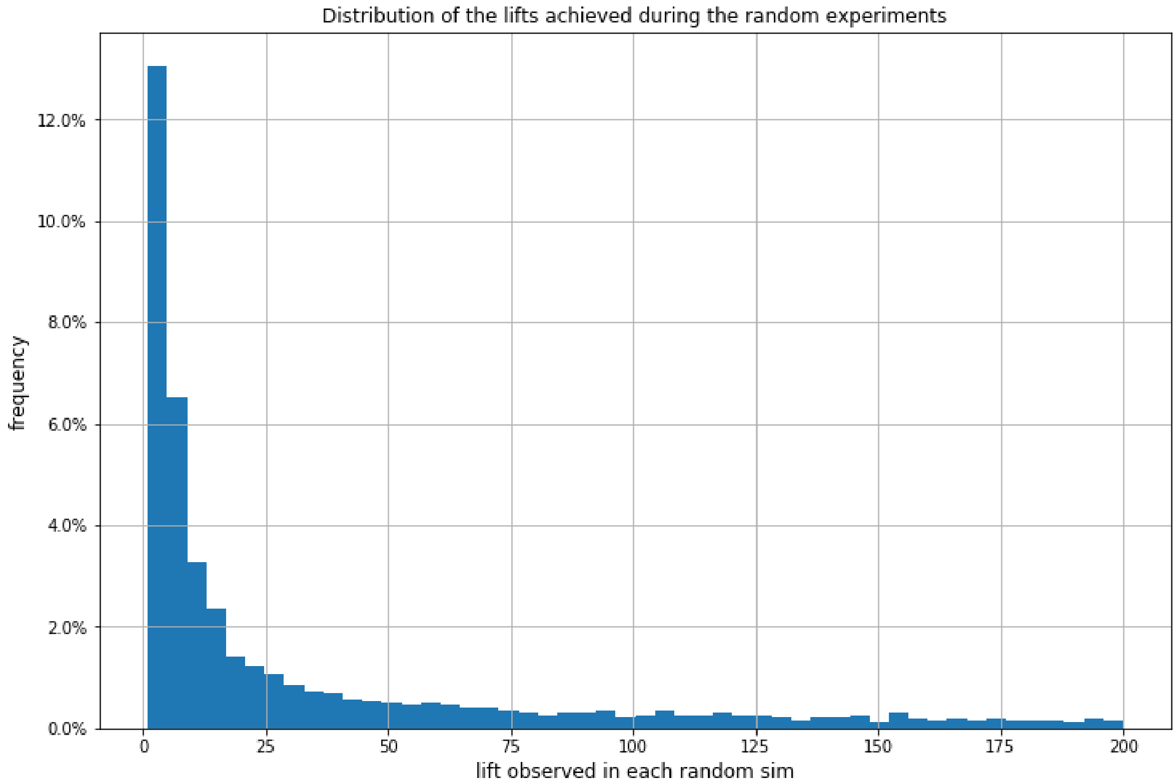


Figure 3: Distribution of the observed lift in each random simulation.

5 Conclusion

As seen from Fig. (2c), by achieving an almost equal cosine similarity score as the BAU, we are achieving a higher value (Fig. (2b)) using the value-based method.

Hence we think that it is worth implementing this value-based recommendation method into a live A/B test with the BAU.

6 Miscellaneous

Some questions:

1. Any reason why the articles are so differently valued?
   - I.e., why are there different articles which are quite similar to a given current article, but with so much variance in the values? So much so that we see a big value lift?
2. Do we have any idea how seasonality affects the volatility of article value? Do we have enough confidence on the value being stable over time?
   - Is there a way we can get the c-values within a desired time window, instead of an all time historical value? Yes. But can we also get to plot the values over time?
   - Can we somehow test these c-values to ascertain our attribution of an article to a value?
3. In the process of showing article an A instead of article B, I hope that we aren't just transferring money from one pocket to another?
4. [Do analysis] It would be good to know how often "high-value" pages are recommended with this approach vs the similarity based approach.