1. Explain the purpose of the algorithm in your app. Clarify whether the chosen algorithm is the only existing algorithm that can perform the intended task. List alternative algorithms and explain your choice.

Note: Here we discuss the algorithms used in our forum post recommendation system. Other central algorithms like search and sort have been covered in Milestone 1.

A core feature of our forum post is the recommendation page based on prior user interactions with posts (likes, comments and saves). The assumption is that the post a user interacts with is representative of their present needs and circumstances. This is especially useful for inexperienced or expecting parents who do not know what they do not know. A curated page may expose them to issues they have not anticipated and the best approaches they have not considered. Furthermore, it enhances user experience: users do not have to scroll through irrelevant posts.

Is this the only algorithm that can perform the task of recommending forum posts to users? Certainly not it is quite a simple one which makes use only of data regarding direct interaction with posts (ie likes and comments) grouping similar users, rather than something used in other social media like Tiktok which considers how much time a user spends viewing a certain post, recommending similar content accordingly. However, a deep learning algorithm does not fit the function of our app: we do not intend to monetise the app hence there is no reason to keep a user on the app as long as possible. Furthermore, it would be too costly and require too much work.

Another algorithm we considered was clustering algorithms, such as k-means or categorical clustering. However, clustering algorithms are more proficient in finding patterns in numbers, not text and categories which is what our forums have. Wherein Kmodes clustering algorithm may be useful in clustering categorical attributes, it does not take into account user data and history.

This can be resolved by using content-based and collaborative filtering may even end up grouping users with children of similar ages based on the posts that they interact with, so we wouldn't need an additional algorithm, like clustering, which actually groups users accordingly.

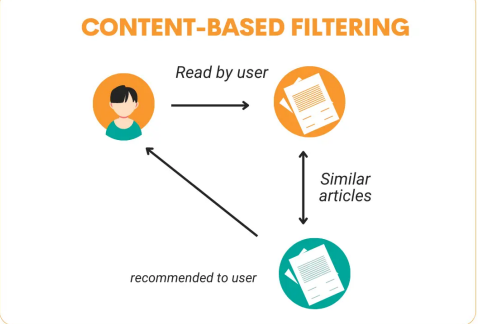
To conclude, we want to keep our interface and user experience simple and manageable, it should be partially curated to the user, especially in terms of the age of their children, but besides that, we want to provide users with access to any information that is available to them.

2. Explain how this algorithm works- what are the inputs and outputs, what is the sequence of steps performed on this input and leading to the output?

In order to provide users with an efficient and functional forum page we will be applying a hybrid recommendation system involving two types of algorithms: content-based filtering and collaborative filtering.

➢ Content-based filtering

This form of filtering analyzes the content of the posts, using item features to recommend similar posts based on keywords, topics, or other content-related features. These are indicative of what a user likes. Natural Language Processing (NLP)



techniques can be used to understand the content and recommend similar posts to the user.

In order to make relevant recommendations, we can use a similarity metric like dot product, using it to score each item according to similarity metrics. These recommendations will be specific to one user and the model will not use information about other users. Some important components in building this recommendation system are the user matrix, item matrix, user-item matrix and features.

Below is an illustration of how the dot product metric could be used in our app.

Sample posts:

- Post A: my child keeps waking up in the night and jumping around
- Post B: I am not sleeping enough because of my children
- Post C: can I feed my 1-year-old carrots?
- Post D: my 6-year-old hates school food, then how
- Post E: sugar makes my 8-year-old super crazy and energetic
- Post F: can I put my 1-year-old in preschool?

User matrix (or vector as it is only one user):

	Post A	Post B	Post C	Post D	Post E	Post F
User interaction *	1 (only liked)	7 (liked, commented and saved)	3 (liked and commented)	6 (commented and saved)	**	**

* These are 'ratings' (like = +1 'point' comment = +2, save = +4), weighted based on perceived importance
**not seen by user yet

Item matrix (binary coded)

Post	Food	Sleep	School	Hyper
A	0	1	0	1
B	0	1	0	0
C	1	0	0	0
D	1	0	1	0
E	1	0	0	1
F	0	0	1	0

User-item matrix:

Post	Food	Sleep	School	Hyper		User rating
A	0	1	0	1		1
B	0	7	0	0		7
C	3	0	0	0	→	3
D	6	0	6	0		6
E	0	0	0	0		-
F	0	0	0	0		-

Sum of each topic	9	8	6	1
Norm sum of each topic	0.375	0.333	0.25	0.042

Now based on these matrices we are able to make find the dot product between two topic values and the user feature vector can be calculated, giving us recommendation values.

Recommendations:

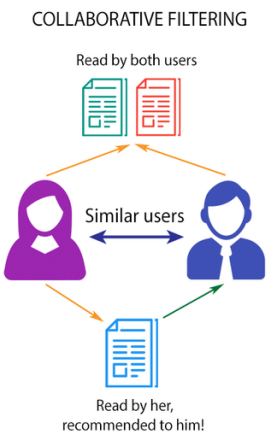
Post	Food	Sleep	School	Hyper		Recomm. to user
A	0	0.333	0	0.042		-
B	0	0.333	0	0		-
C	0.375	0	0	0	→	-
D	0.375	0	0.25	0		-
E	0.375	0	0	0.042		0.417
F	0	0	0.25	0		0.25

Here we see that Post E is more likely to be recommended to the user.

➢ Collaborative filtering

User-based: This form of filtering recommends posts to a user based on the preferences of similar users. If user A likes posts X, Y, and Z and user B has similar interests, posts liked by user B that user A hasn't seen might be recommended to user A.

Item-based: Recommends posts similar to the ones a user has interacted with. If a user likes post X, then other posts similar to X might be recommended.



For collaborative filtering, we will use very similar matrices to the ones used in content-based filtering, just with several users instead of one.

Steps involved in this algorithm:

1. Finding similar users or items
    a. such as by using matrices and similarity algorithms
    b. This would make use of user-item ratings like in the table seen in content-based filtering above (but with more than one user)
2. Predict the ratings of items that are not yet rated by a user
    a. This is based on similar users' ratings of a post, this uses a specific mathematical formula:

$$R_U = (\sum_{u=1}^{n} R_u)/n$$

3. Use the similarity factor to calculate weighted averages, giving a good prediction of how relevant a user will find a post to be
    a. $$R_U = (\sum_{u=1}^{n} R_u * S_u)/(\sum_{u=1}^{n} S_u)$$
4. Make recommendations accordingly!

3. What is (are) the limitation(s) of this algorithm(s)?

Limitations of a content-based algorithm are as follows

1. It can create a content bubble and does not take into account the age of the child and its changing needs as they age.
2. This does not ascertain the quality of the post itself; only its relatedness to similar posts the user interacts with.
3. It does not provide a diversity of content.

Limitations of a collaboration-based algorithm

1. The user may be matched with a user with multiple children, one of whom should be a similar age. The user may be searching content for children who are not the same age and thus recommended post would be irrelevant.

For both algorithms, there is a cold start problem, where they need to be content for it to work. Thus, to resolve the limitations of both algorithms and leverage the strengths of both, we will be adopting a hybrid model. Furthermore, we will be introducing the app in maternity groups; of expecting parents to begin populating the app. Considering these limitations will be crucial to improving the quality and robustness of our solution, providing more effective and useful content for users.

Sources:

Content-based Filtering | Recommendation Systems. (n.d.). Google Developers. https://developers.google.com/machine-learning/recommendation/content-based/basics#:~:text=Content%2Dbased%20filtering%20uses%20item

Rosidi, N. (2023, February 16). *Step-by-Step Guide to Building Content-Based Filtering*. Www.stratascratch.com; Stratascratch, LLC. https://www.stratascratch.com/blog/step-by-step-guide-to-building-content-based-filtering/

Abhinav Ajitsaria. (2019, July 10). *Build a Recommendation Engine With Collaborative Filtering*. Realpython.com; Real Python. https://realpython.com/build-recommendation-engine-collaborative-filtering/