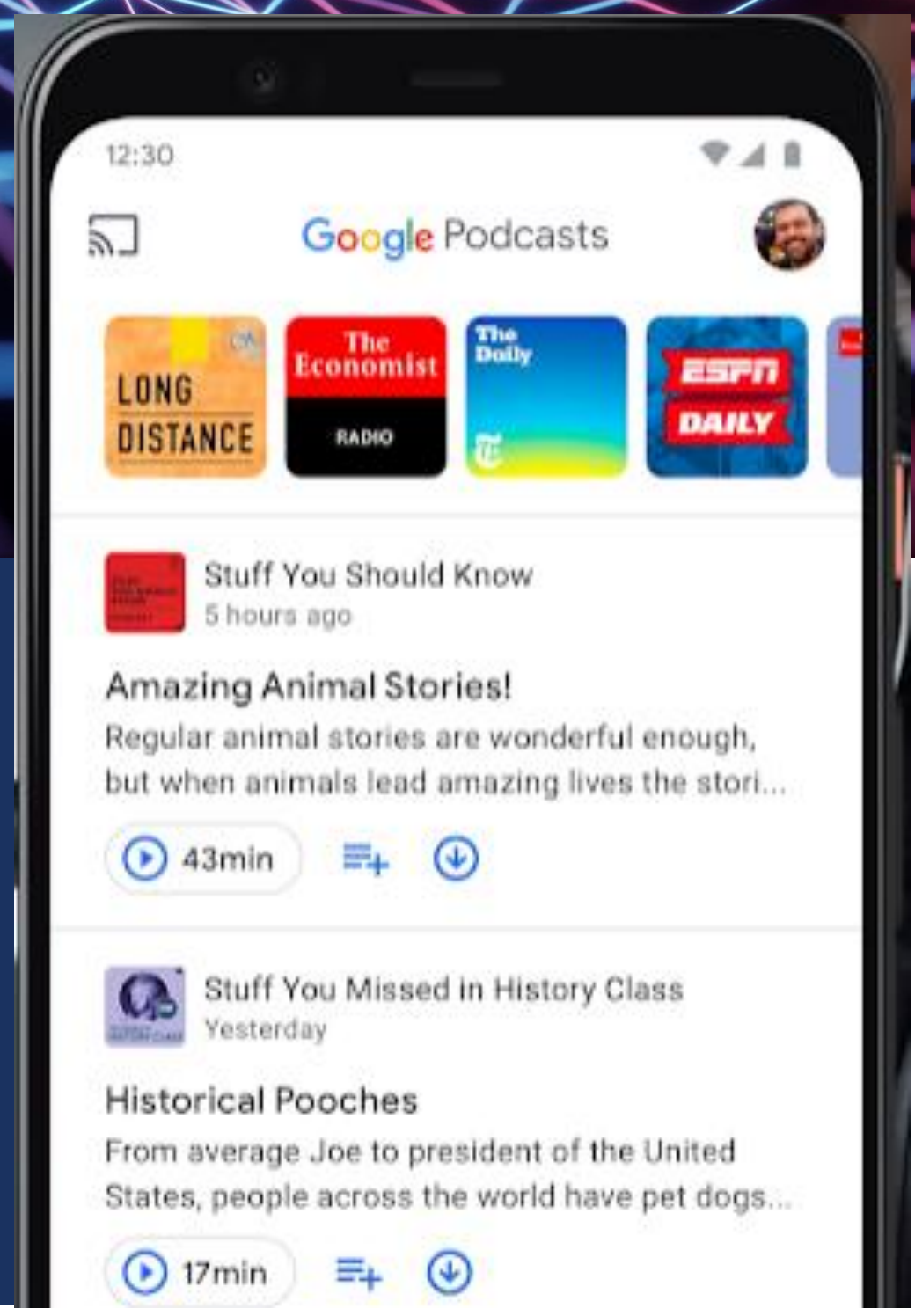
PODCAST RECOMMENDER SYSTEM (FOR AGRAHYAH TECHNOLOGIES)

BY PINAKEE KAUSHIK , PHONE NO:-9599776701, IAMPINAKEE2098@GMAIL.COM

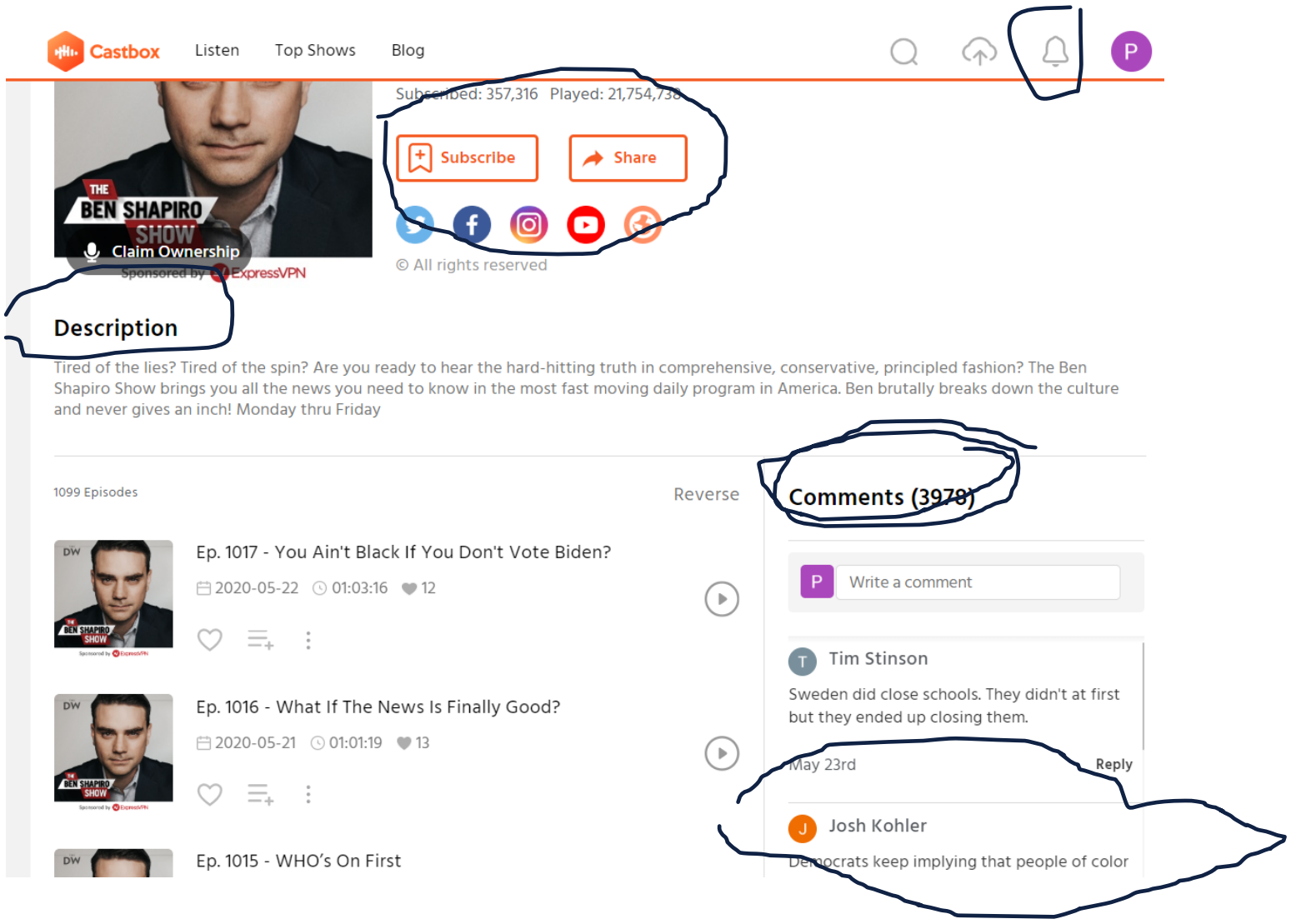


WE WILL BUILD OUR RECOMMENDER SYSTEM BY COMBINING 4 APPROACHES

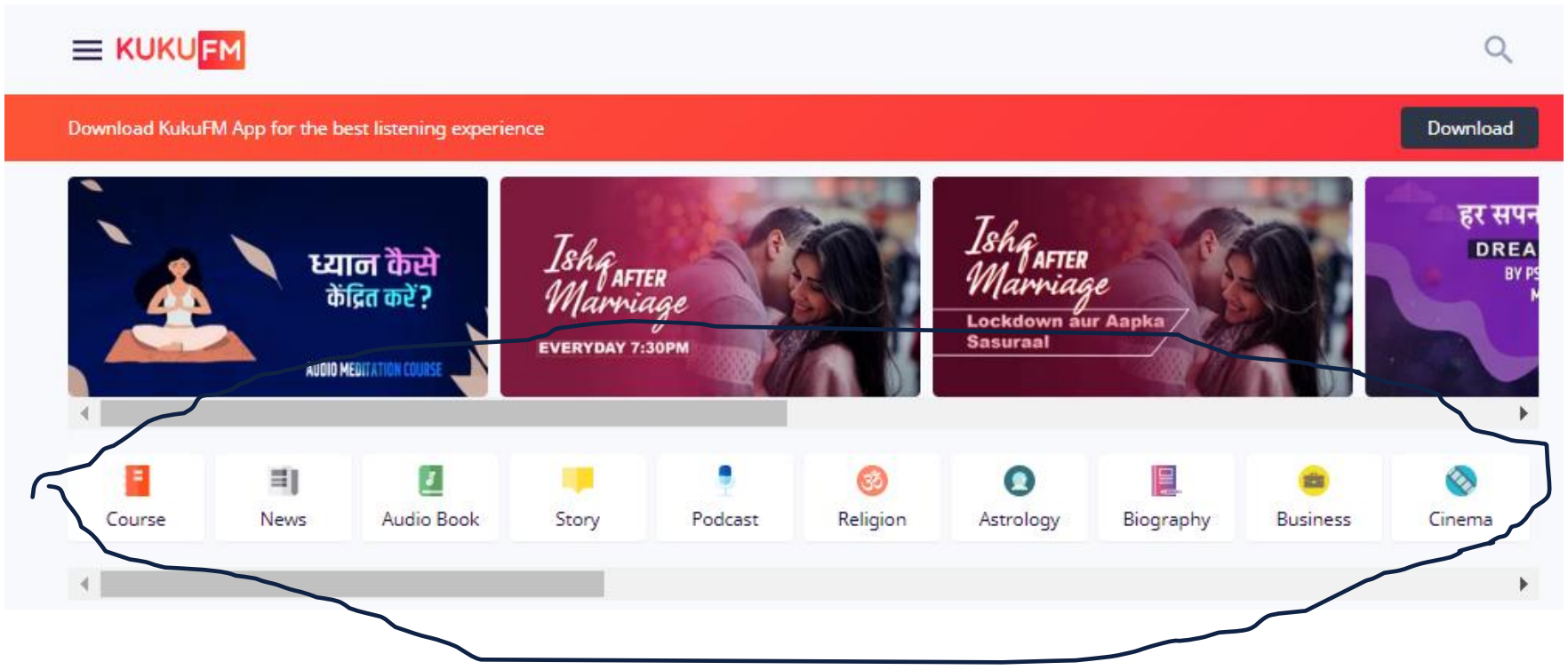
- Collaborative filtering
- Content based filtering
- Multi-Armed Bandit Framework
- Additional methods

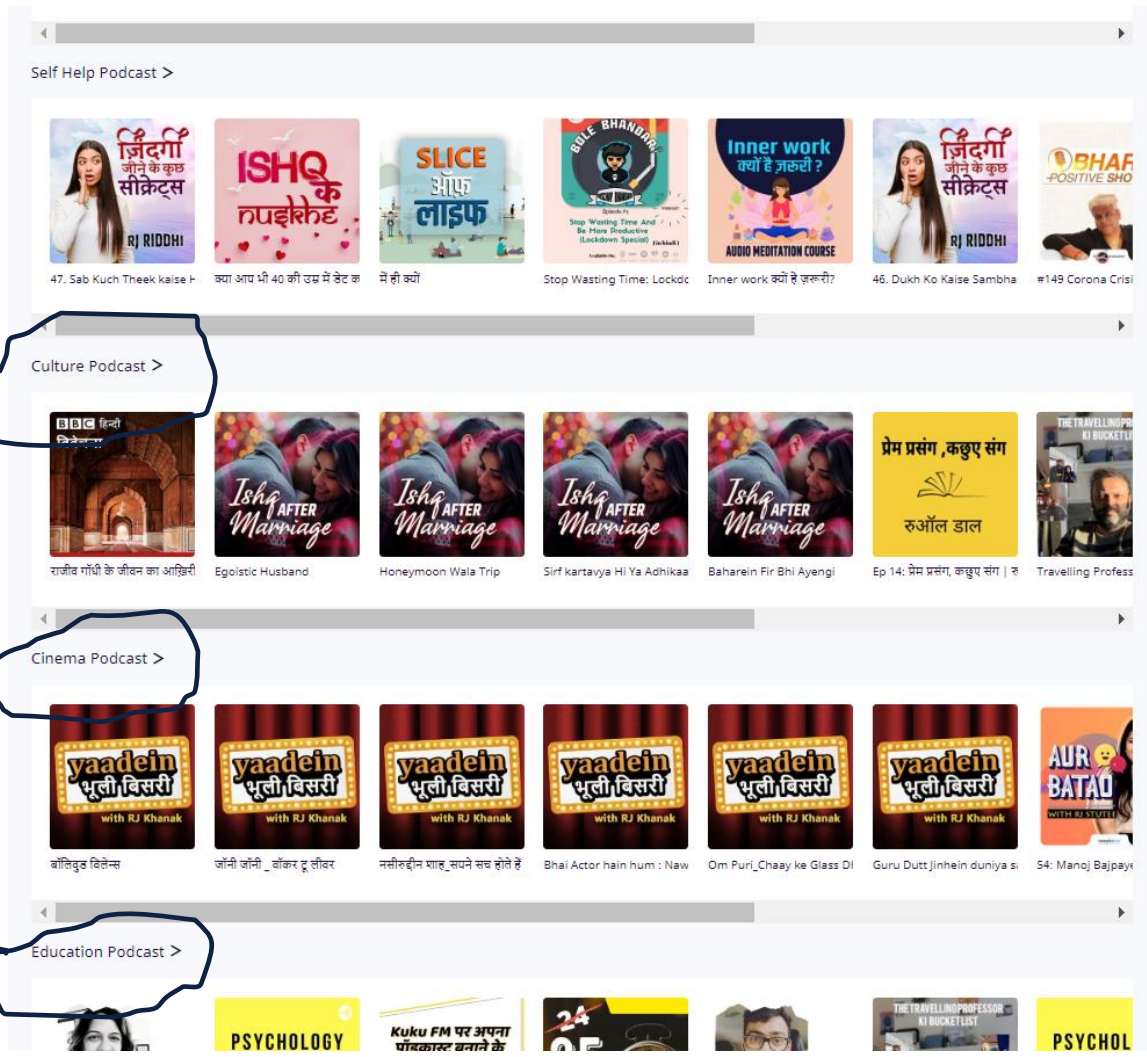
FIRST WE WILL LOOK WHAT DATA AND FUNCTIONALITY WE CAN USE WHICH CAN ACT AS FEATURES FOR OUR MODEL

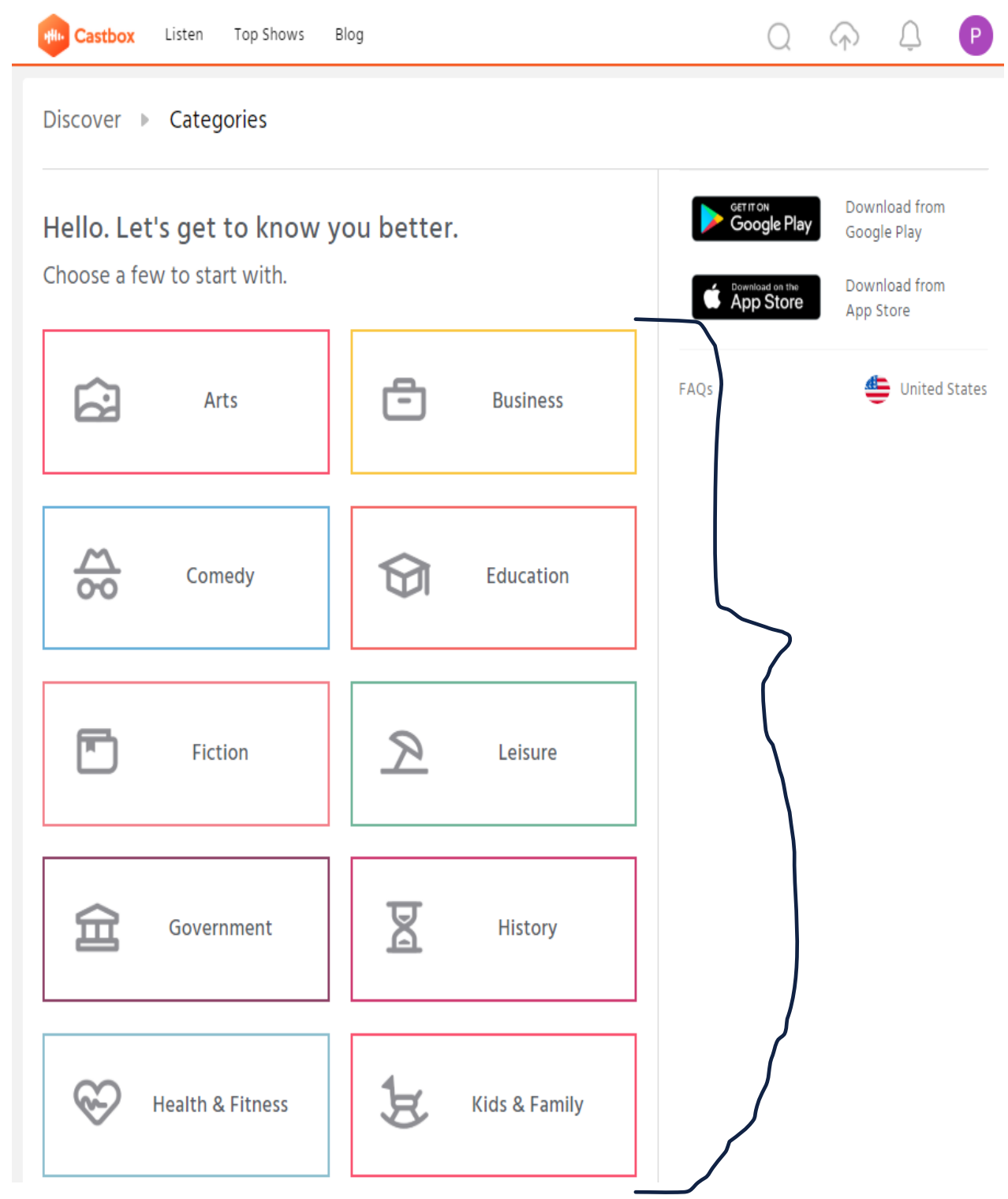
Likes, comments, description of Podcast, share, Skip , subscribe features of apps shared to me like Castbox



Categories features.







When we start to build our recommendation model we will have all these data and we will use this data only as our features , let us look at what fields we have in the data base of each user:

Like , comment , skips in the video, ratio of duration of podcast listened, channel subscribed or not and shared.

A noob and starting approach (Using collaborative filtering on likes)

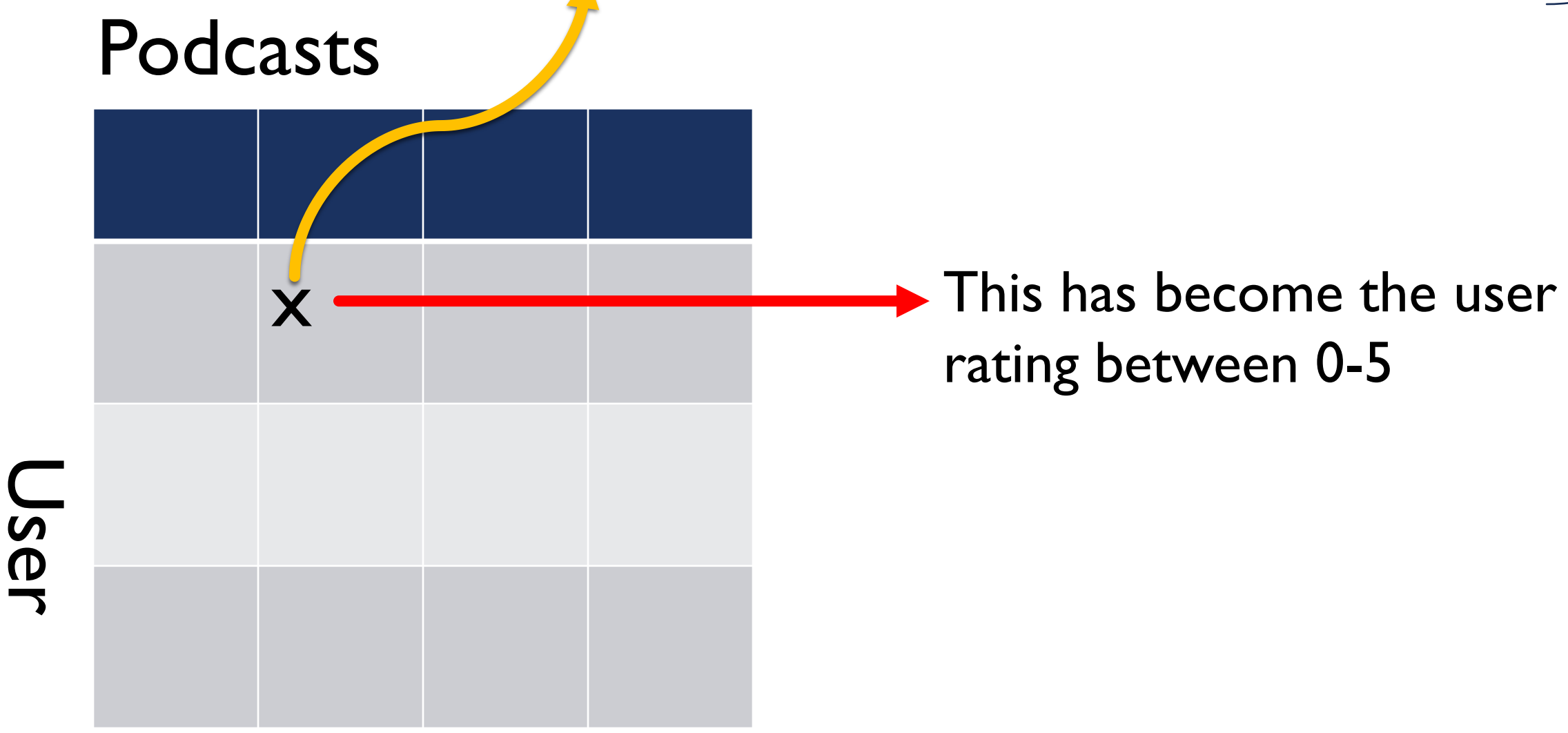
Podcasts

Users					
	I	I			
I		I	I		I
	I	I	I		
	I			I	

A better approach (Combining all the user's features mentioned above to build ratings)

Lets combine mathematically all the user features:

α*Liked or not + μ*ratio of audio listened to total length of audio + ß*Sentiment of the comment + ý*podcast shared - þ*how many times audio skipped + k*listened more than one time+c*Subscribed



Explanation: Here alpha , beta , gamma etc are all the constants which are decide based on the business rules and A/B testing that will tell which weight of the parameter depending on how much impact that parameter have in determining the how much user liked / disliked a particular podcast. For example ratio of audio listened to the total length of audio can be a factor in determining and we can give it weight accordingly. Similarly how many time user skipped the audio, it has although very little determining factor but can be considered.

For example lets just assume for now after getting the domain knowledge we give the constants the value as follows. The matrix will be formed from the database and these parameter only:

α*Liked or not + μ*ratio of audio listened to total length of audio + ß*Sentiment of the comment + ý*podcast shared - þ*how many times audio skipped + k*listened more than one time+c*Subscribed

α = {2 if podcast is liked else 0}

μ = {-0.2 , if 0 <= ratio <0.10,
0.2 if 0.10 <= ratio <0.80,
0.9 if 0.80 <= ratio <100 }

Or {1 - log(Time-watched/total time) }

lies between 0 and 1

ß = {0.2 if sentiment score of comment is positive,
0 of sentiment score of comment is neutral,
0 if user didn't comment,
-0.2 if sentiment score of comment is negative}

Ý: { 0.3 if shared else 0.1}

Þ : { 0 if skipped less than 15 times else -0.2}

Note: (This feature may or may not be that imp. We will get know about it after testing)

k : {0.8 if watched more than once else 0}

Note: The above feature tells a lot about the user taste hence after domain knowledge it got higher rating}

c: {0.8 if the podcast user is watching is part of subscribed channel}

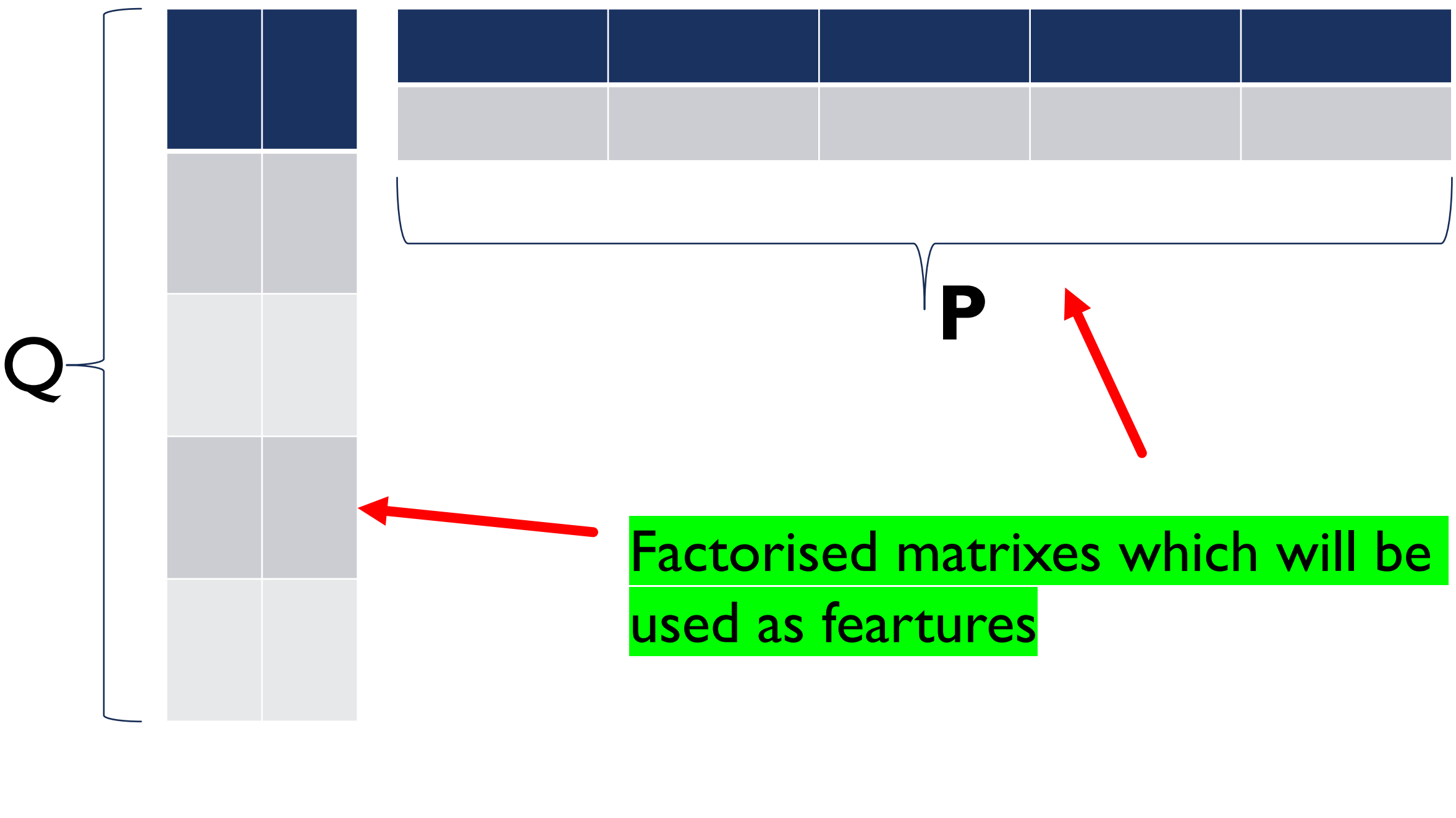
Adding all the max values give 5 rating

Matrix Factorization using SVD++

0.4		3.1		
	4.8		2.7	
	4.5	1.2	4.1	
				0.4
2.6			3.5	1.7
		5		

We will take those constants such that the maximum rating can be interpreted as 5 and minimum 0. The ratings in this case can be any real number between 0 and 5

Example user-podcast matrix

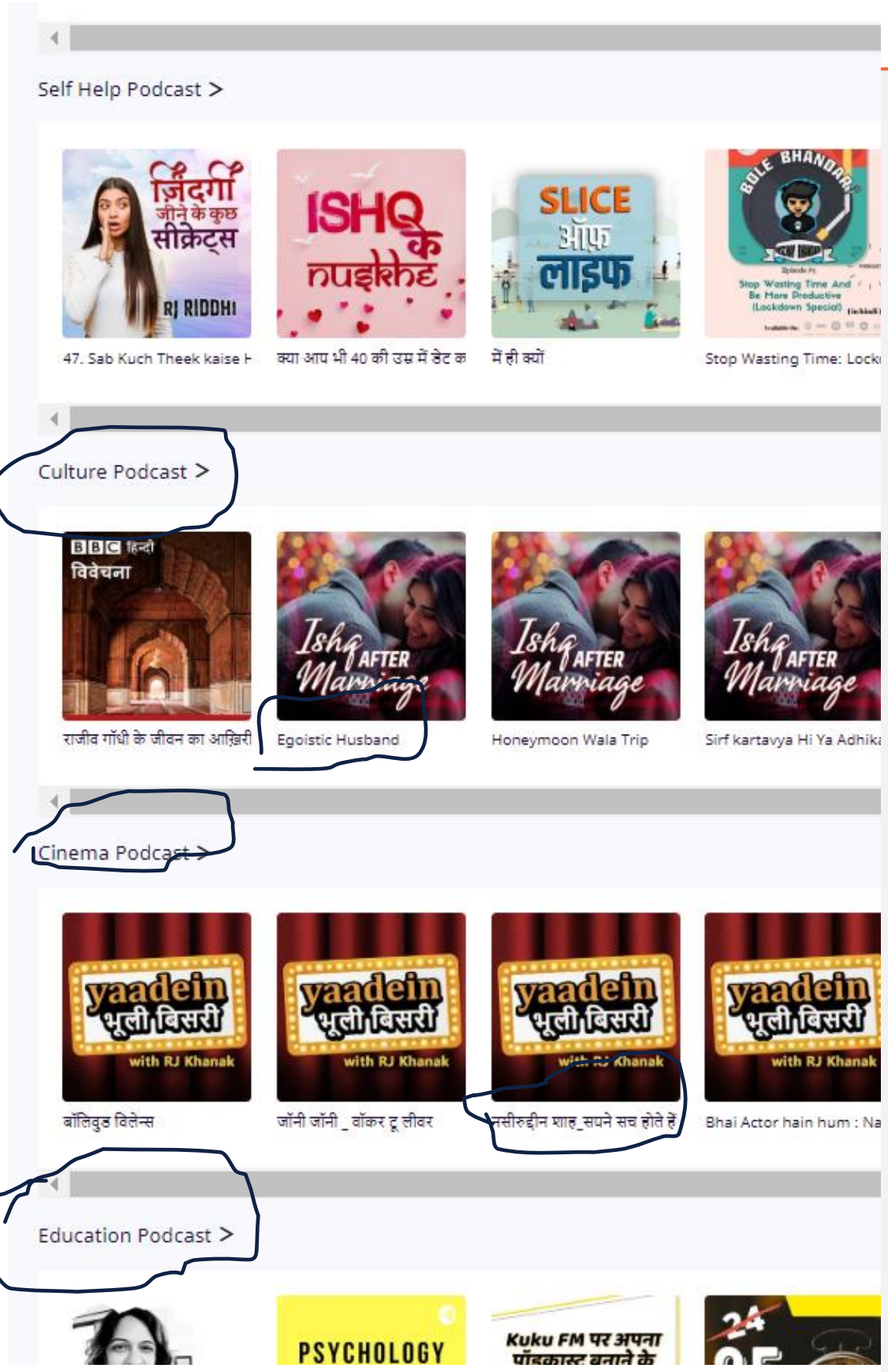


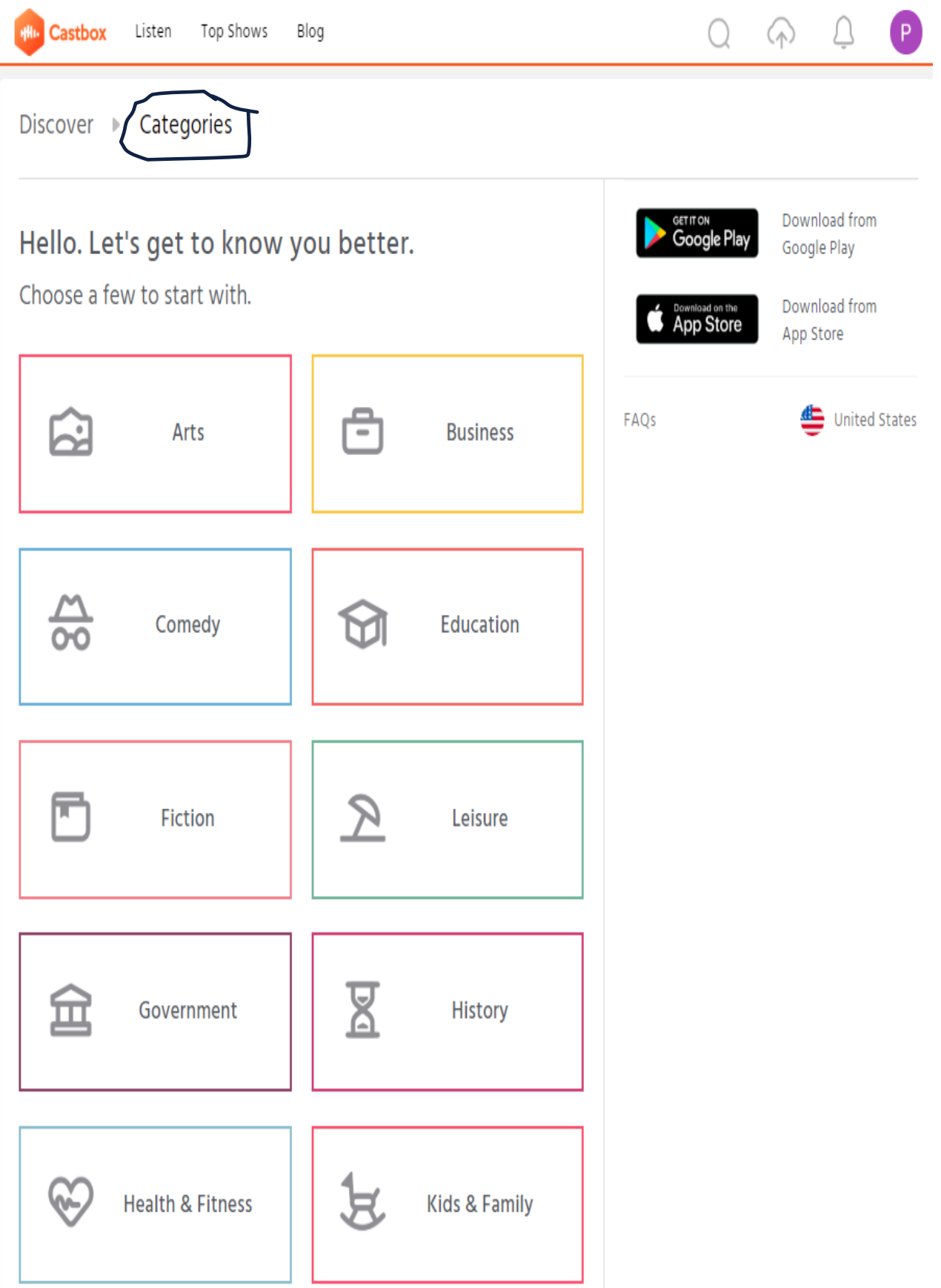
Factorised matrixes which will be used as feartures

SVD is a technique of matrix factorization . Any matrix can be represented by the multiplication of its factor matrix represented by U*∑*V. SVD++ uses implicit feedback technique.

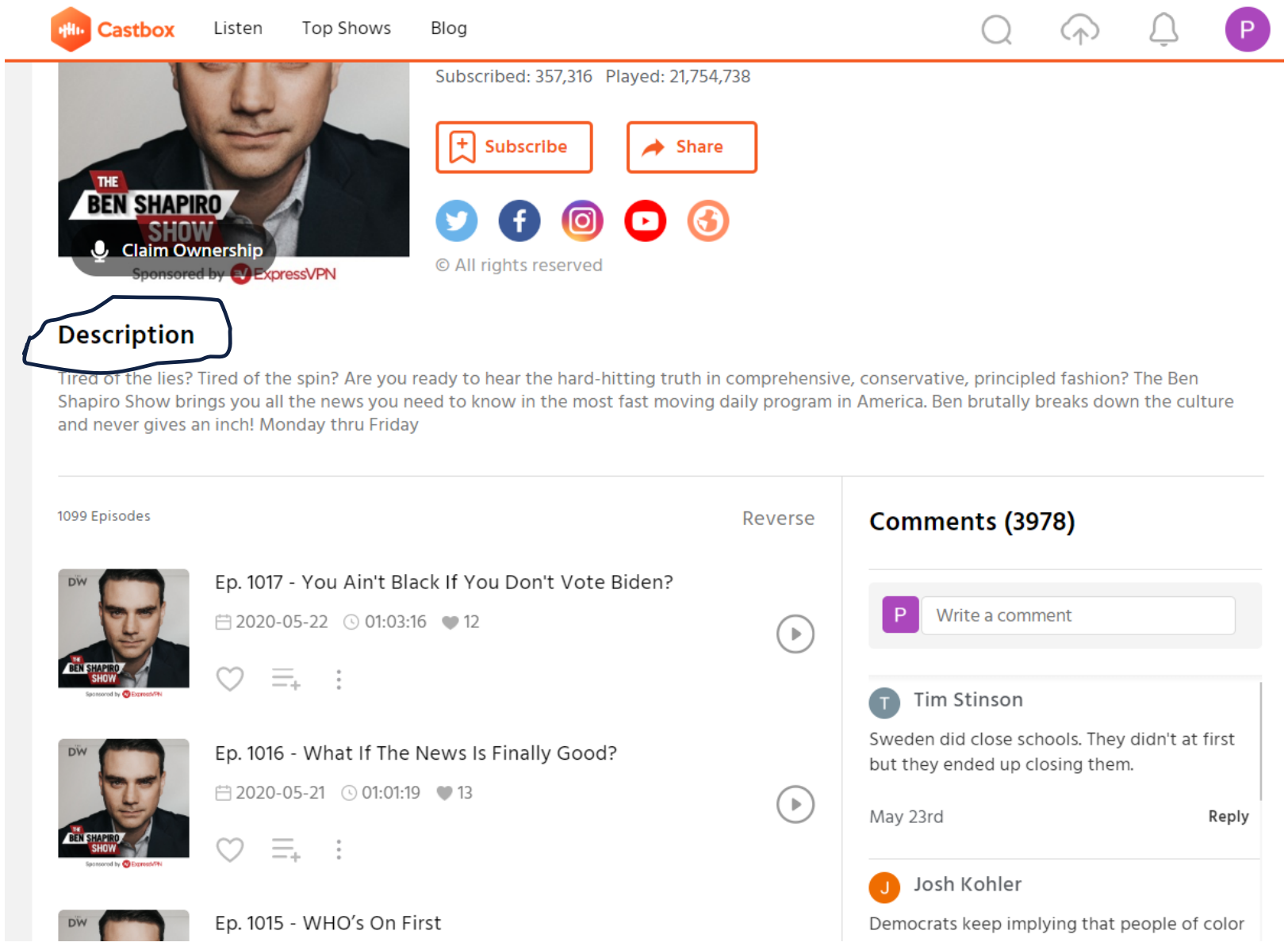
Using content based features:

Let us see what are those: (Categories and titles):





Let us see what are those: (Description):



Out of "Category of podcast, description of podcast, title of podcast, author of podcast and keywords",Category of podcast, description and keywords can be useful. We can do one hot encoding or Response coding on the Category feature and we can extract the most imp keywords as well as the semantic meaning using word2vec from the description.

Possiblity: Due to huge length of bag of words, of dimensions of this feature can become really large, so we have to keep in mind if this feature affects the latency requirement (i.e time taken to recommend in this case.)

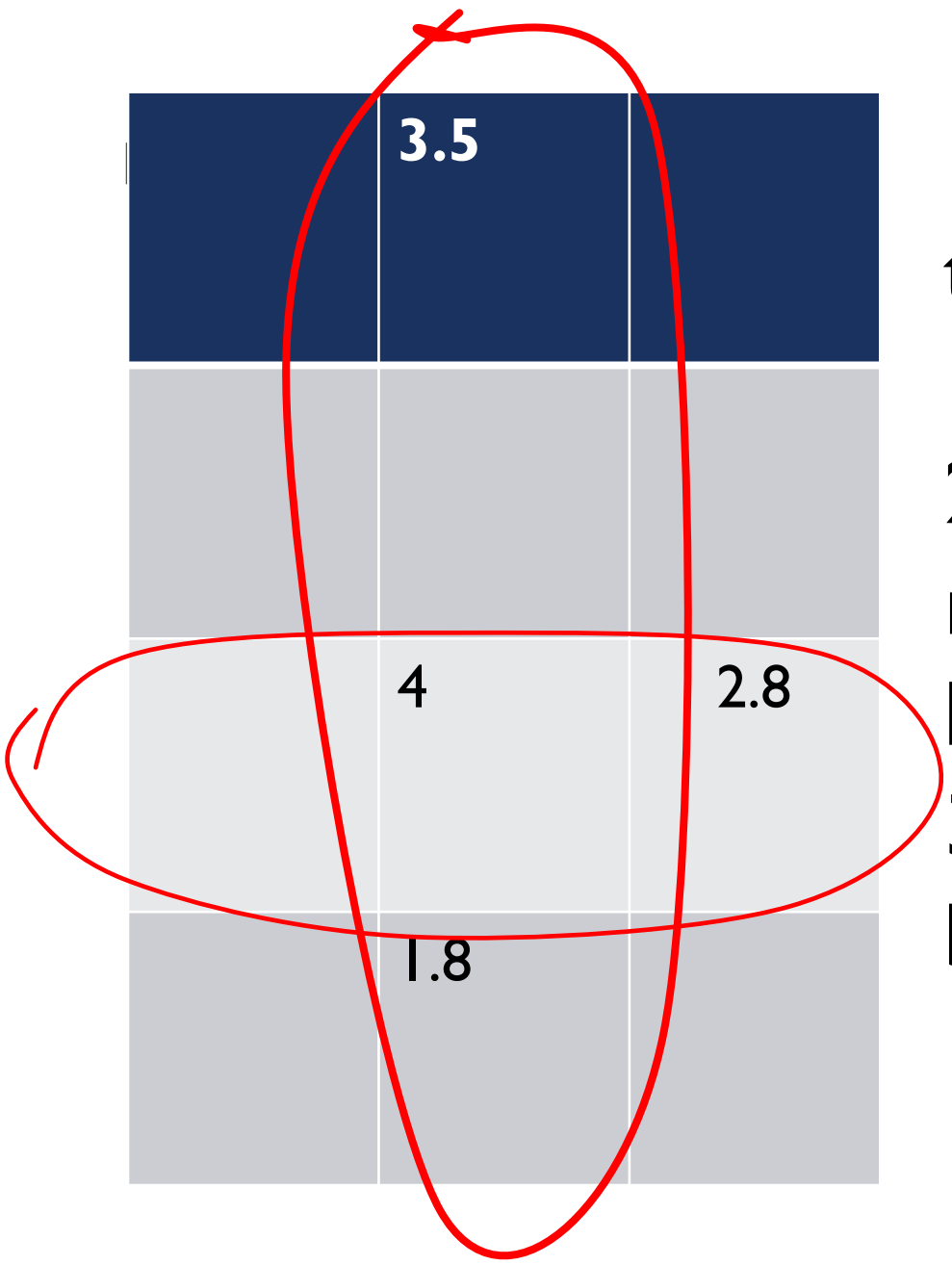
Aim: How to predict the rating given by new user to a podcast he /she has not seen. We can pose this as a regression problem.

<--------Featrues-------→ ←-Rating->

We have to form the features now and predict the ratings and that should close to the actual ratings.

Error Metric : MSE = (predicted rating-Actual rating)^2/N

CREATING FEARTURES:



1> Global average rating: mean of all the ratings.

2> User bias no of ratings(b_u): Avg. rating given by the user to the podcasts he/she has watched

3> Podcast bias (b_i): avg ratings given by user to this podcast.

4> Category of Podcast: One hot encoding

Romance	Motivation	Business	Yoga
0	0	I	0

5> Description of podcast: First taking the bag of words of all the descriptions , and then taking 300 avg word2vec-tfidf fearues.

6> Keyword tags of podcast

7> Feature from matrix factorization:

User bias , globa average, podcast bias, P and Q (as calculated earlier)

(Features)

Model I

Minimize

$$\sum_{i,j}^{K}(R_{ij} - (b_u + b_i + mu + P.Q + C * L2\ regulztion)$$

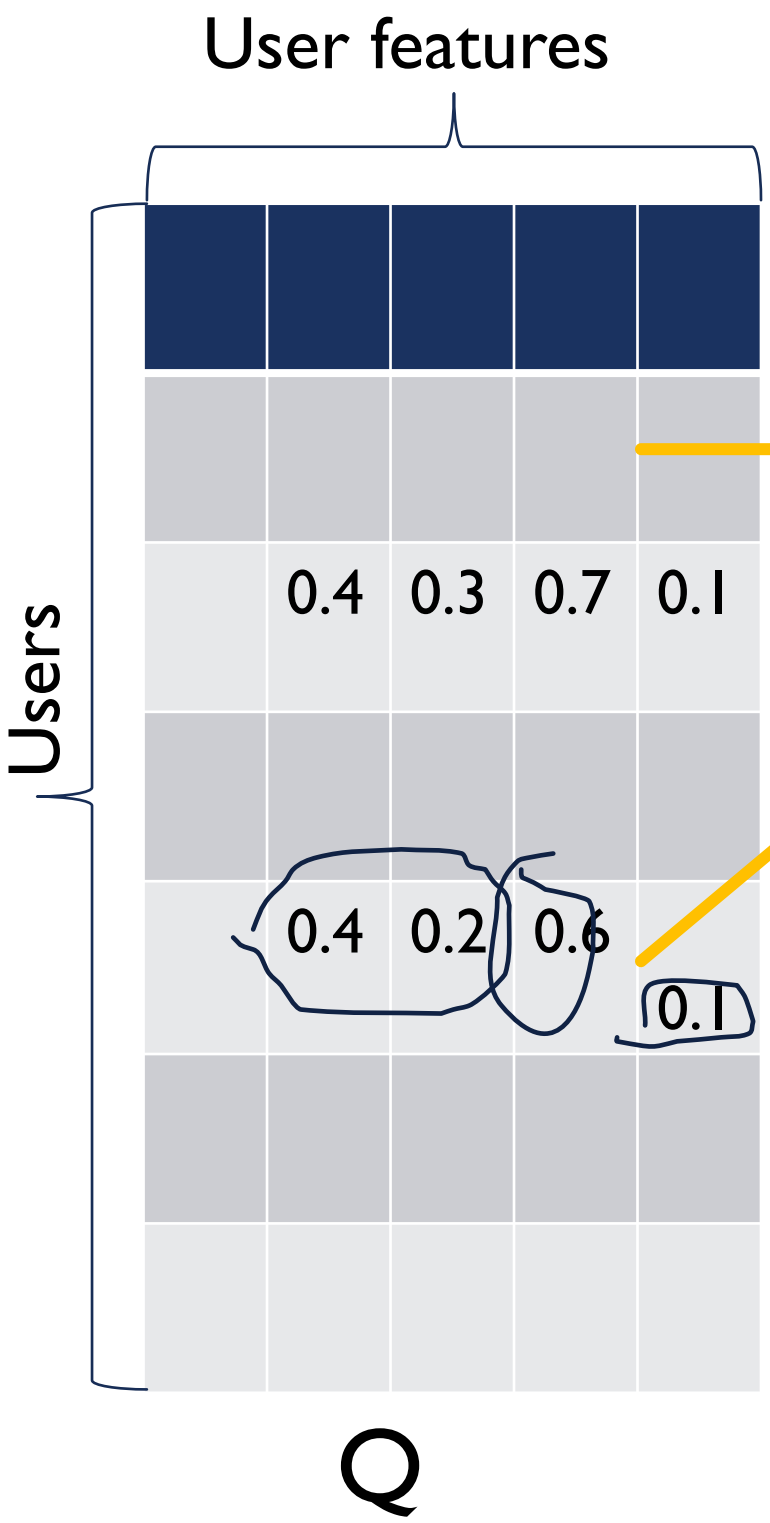
Mse

Storing these parameters and new fearures as predicted user biases and predicted podcast biases

7> Difference of user bias and predicted user bias: That is this feature tells us how much bias difference does this user have compared to the calculated avg. bias

8> Difference of podcast bias and predicted podcast bias: That is this feature tells us how much bias difference does this podcast have for this user compared to the calculated avg. podcast bias.

9> Top 5 similar user's rating for a podcast:



similarity(user i, user i+n) =
cosine(user i, user i+n) =
$\sum$ (ui. ui_n)/ (|ui|^2 * |ui_n|^2)^1/2)

Suppose user_i and user_i_n are found to be most similar, we will take the user_i_n rating for that particular podcast. Similarly we take top 5 such users and take their ratings as features .

9> Top 5 similar podcast's rating for by that user: Same concept as that of feature 8.

Concatenate all features

All the user parameters like Bias difference, similar user's Rating, global avg rating etc. For a particular user id

All podcast parameters like bias, Bias difference, similar podcast's Ratings, for that particulat movie, avg word2vec tf-idf , categories, keyword tags

Rating given by this user to this podcast.

X

Target

Splitting our dataset into train and test

Applying various machine learning models

- Random Forest refressor
- XGBOOST
- Various Ensemble techniques

Performance Metric : Mean squared errors

Hyper parameter tuning

Predict the podcast rating for rest of the user Who haven't rated the podcast yet

Podcasts

Users

Final Predicted ratings

Matrix factorization (Using truncated SVD)

User Matrix

Podcast Matrix

Show recommendations

Because you have watched xyz podcast , you may watch:

Show top 20 similar podcast from the Podcast matrix

AND

Because people watching this pocast also watched these podcasts , you may watch:

Show top 20 rated podcast by these users from the Podcast matrix.

CONCLUSION: WE HAVE BUILT A RECOMMENDER SYSTEM FROM COMBINING THE CONCEPT OF COLLABORATIVE FILTERING AND CONTENT BASED RECOMMENDATIION

Multi-Armed Bandit Framework

When the user first log in for the first time we don't have any information about his tastes, so we start with the cold start strategy called EXPLORATION AND EXPLOITAION. We make use of epsilon-greedy method as tradeoff. So lets take a look what EXPLORATION and EXPLOITAION is:

First we will generate pool of tiles of the podcasts for the user to choose. We define the set of Actions A € {…} which the user might take such as {clicking on podcast, liking the podcast, scrolling through the playlist etc} and for action there is corresponding award.

Random pool of tiles and the user will choose from them and take action. For each action there is a award. Our objective is to maximize the award. In reinforcement learning we refer to it as q value. We cant just put Content, we also need to exploit the results we got in the form of rewards. This tradeoff between two is maintained by a threshold values using the epsilon greedy method.

After enough of explorations, we will keep finally store user and content parameter having the highest reward and then we will start to predict the rating for the user and recommend podcasts.

Additional Methods which I am learning in detail like these concepts, would love to learn and work on it with **Agrahyah Technologies**

1)

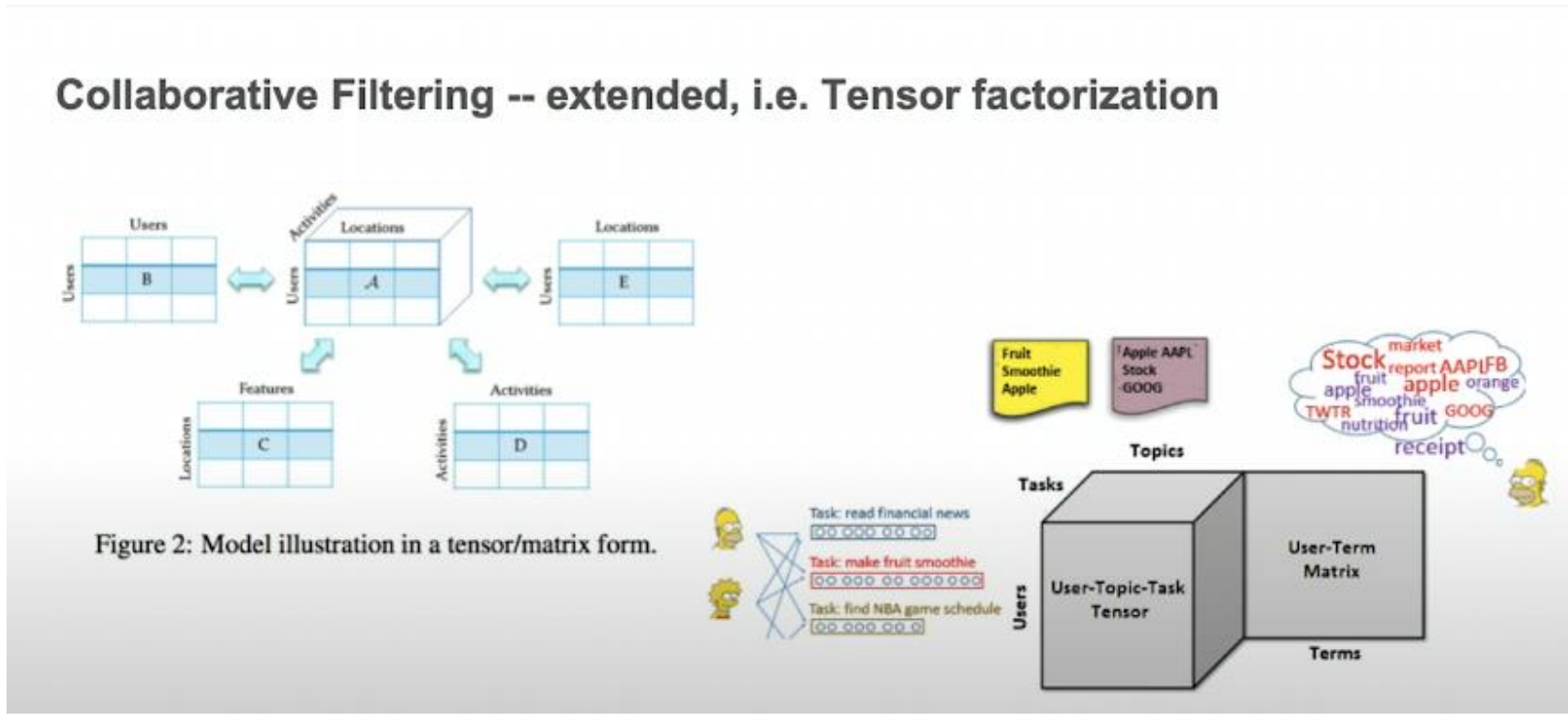
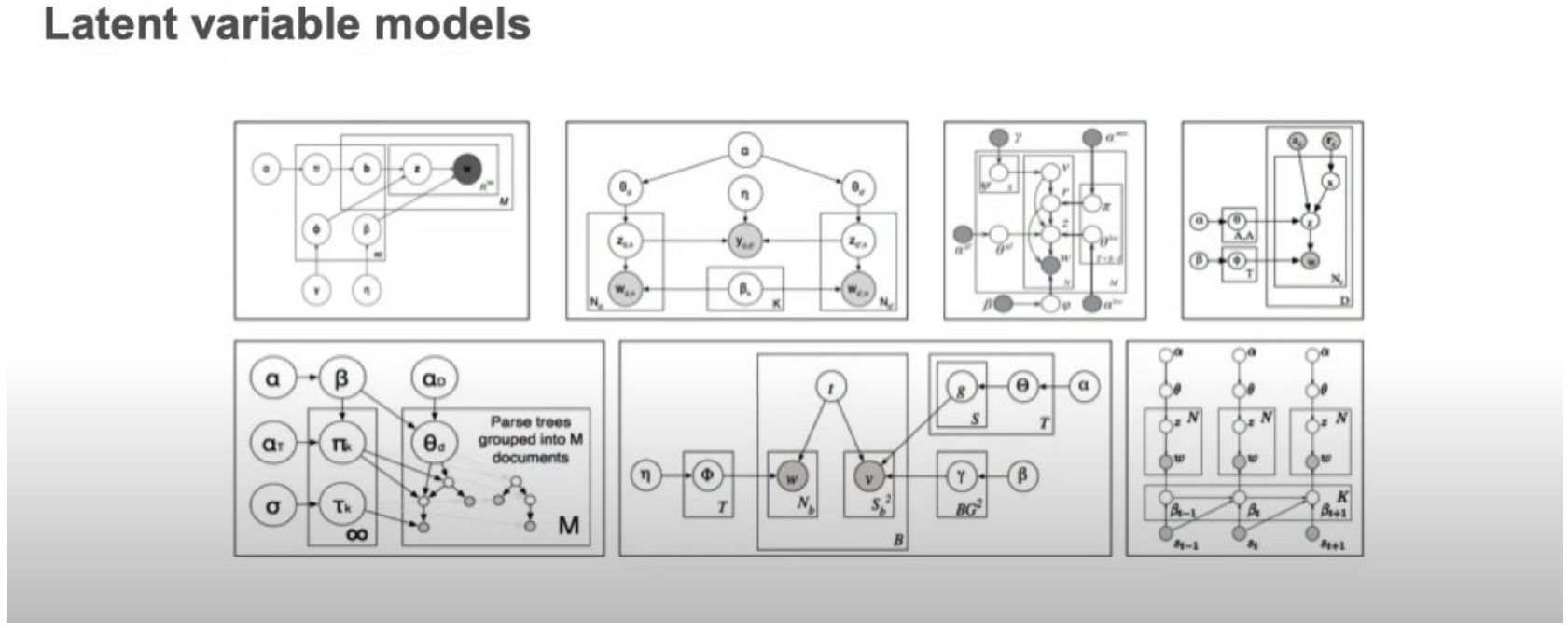
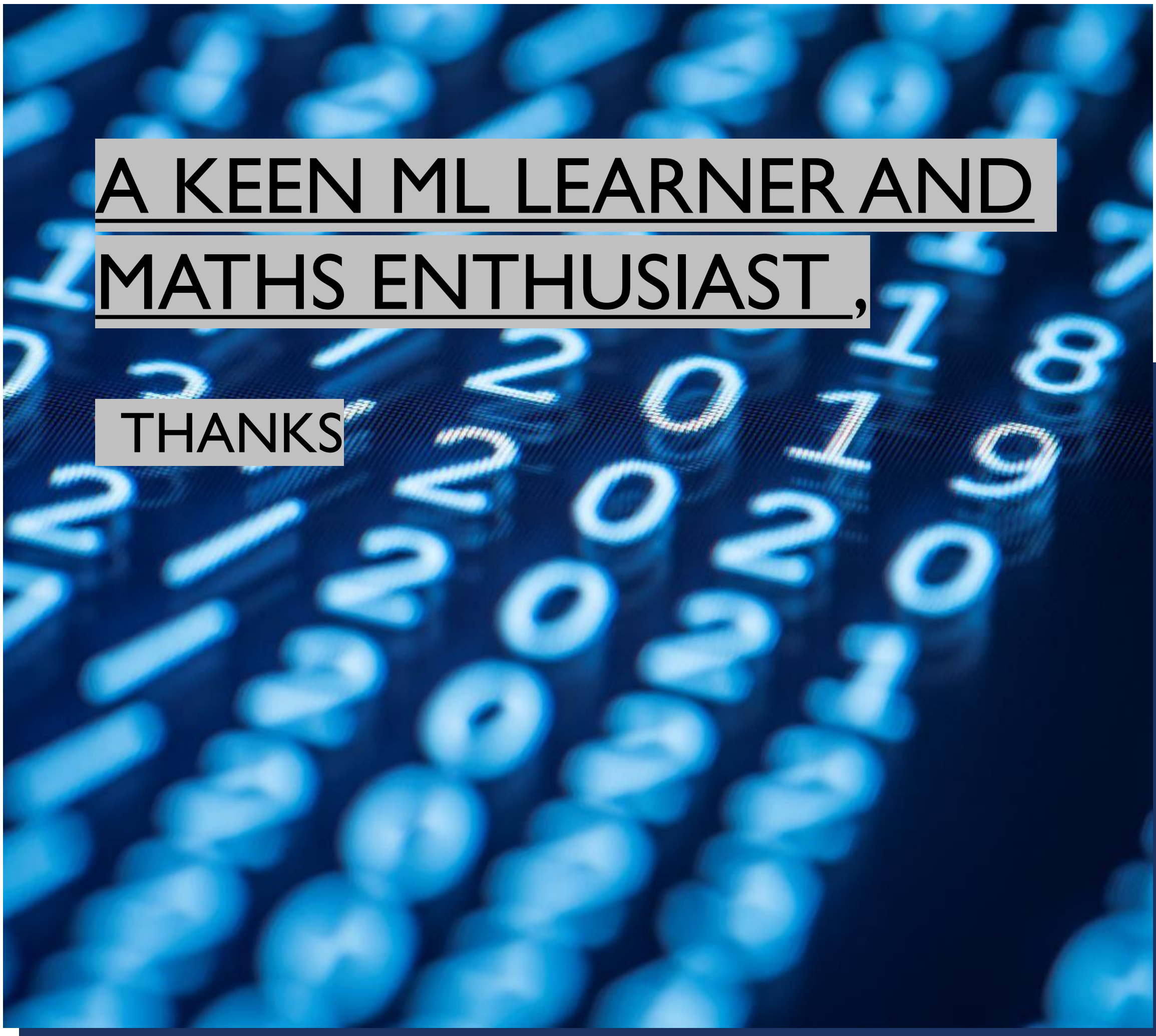


Image source: https://www.youtube.com/watch?v=KoMKgNeUX4k&t=77s

2)




A KEEN ML LEARNER AND
MATHS ENTHUSIAST ,
THANKS