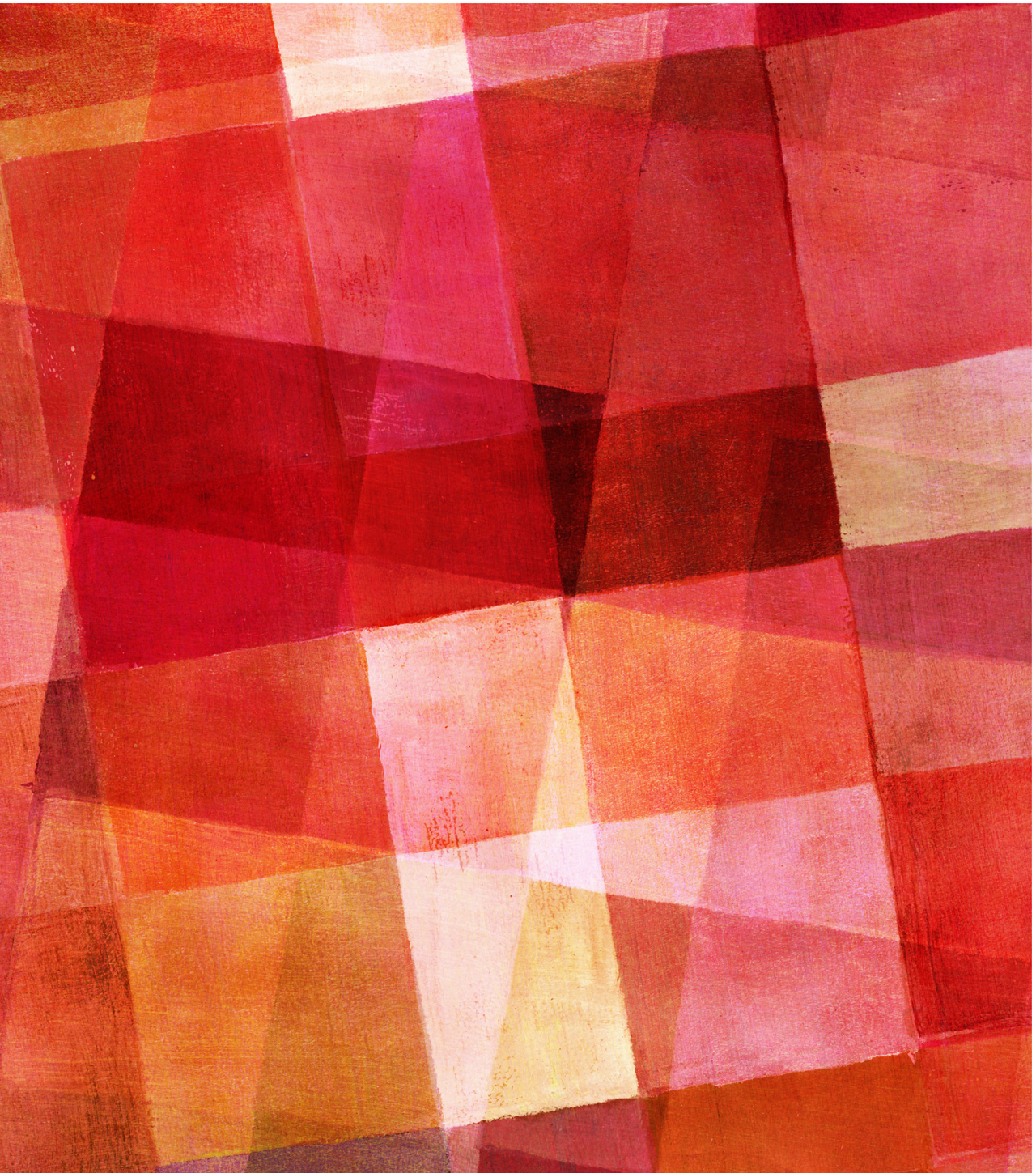
AIRS

ARTIFICIAL INTELLIGENCE RECOMMENDER SERVICE

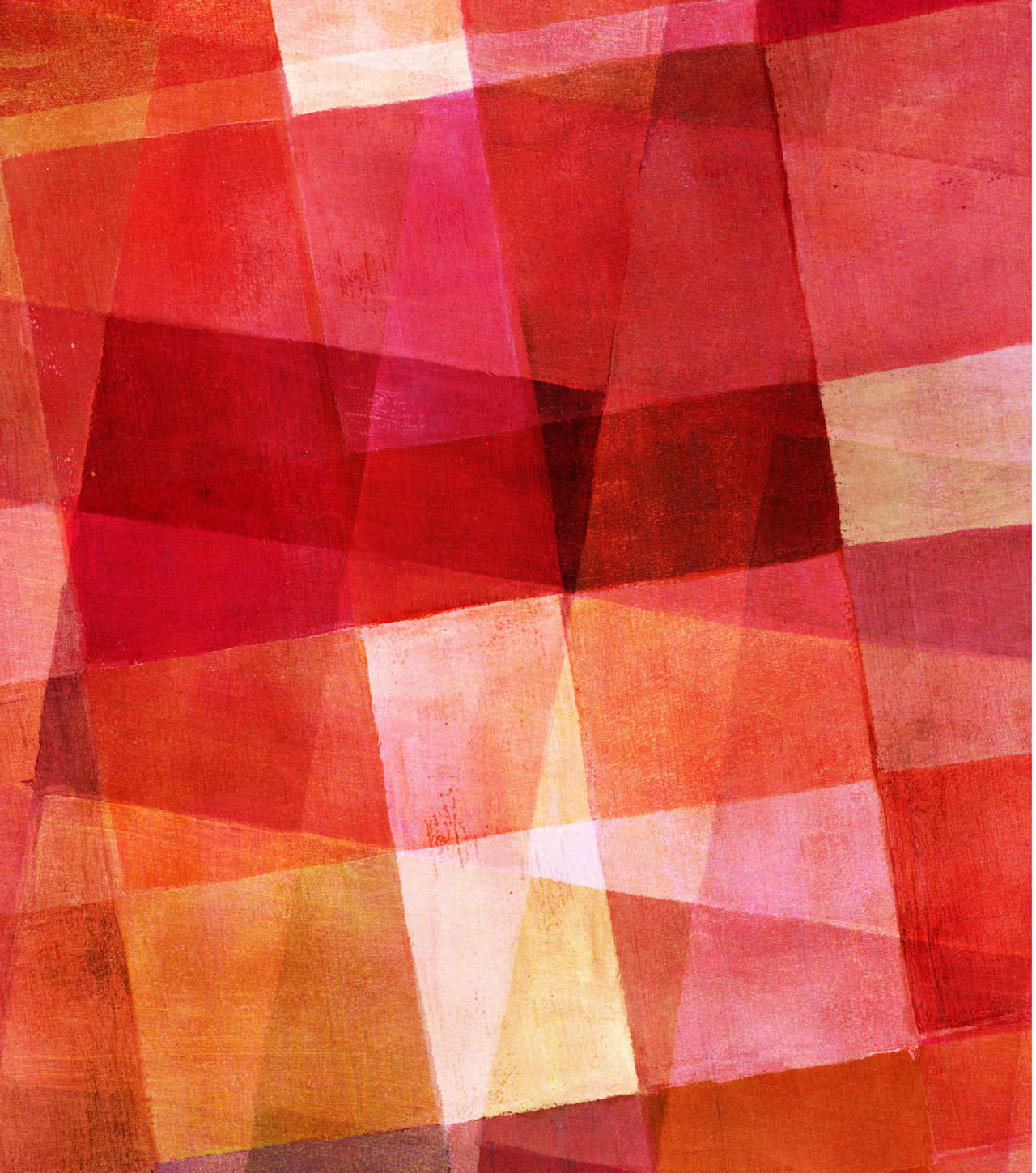


AGENDA

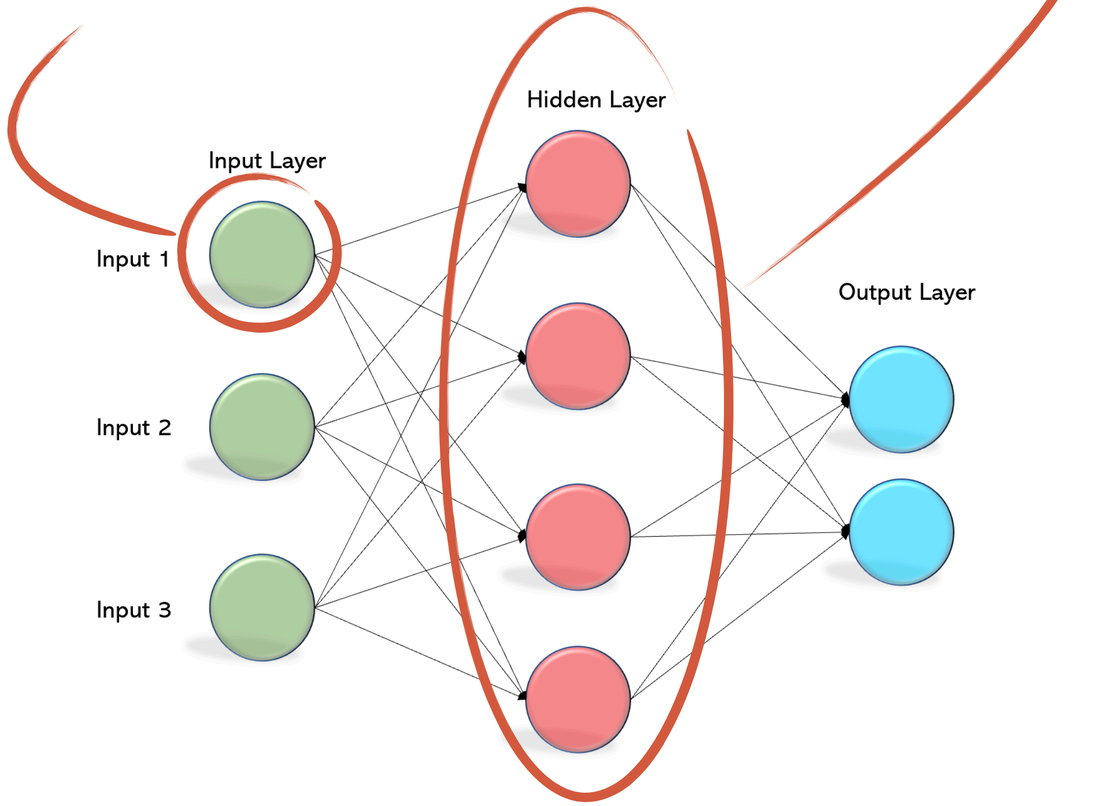
- What is a neural network
- How do we build recommender service
- Proposition
- AIRS
- Q&A

WHAT IS A NEURAL NETWORK

It is a method of finding the relationship between a collection of inputs and its corresponding outputs

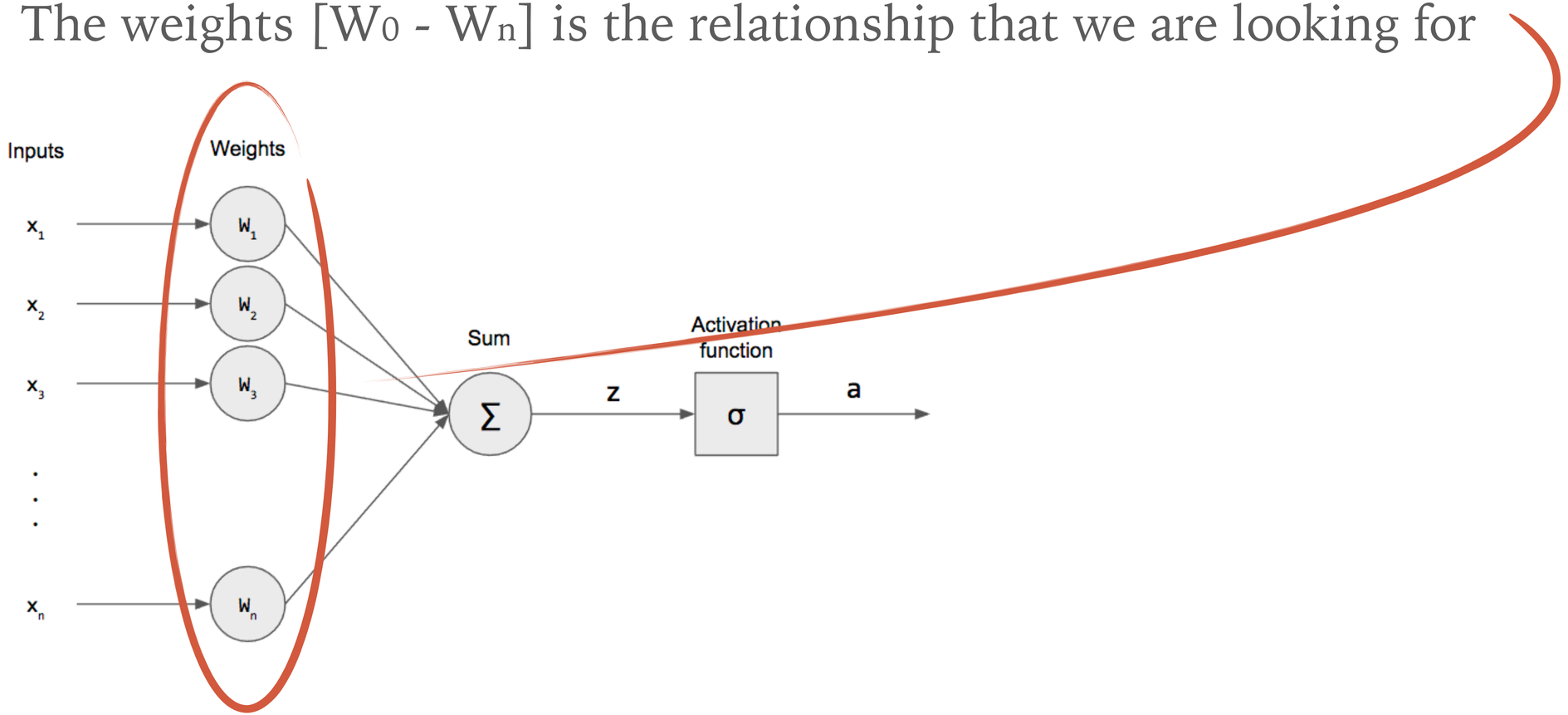


- It is made up of 1 or more layer
- Each layer is made up of 1 or more neuron (perceptron)



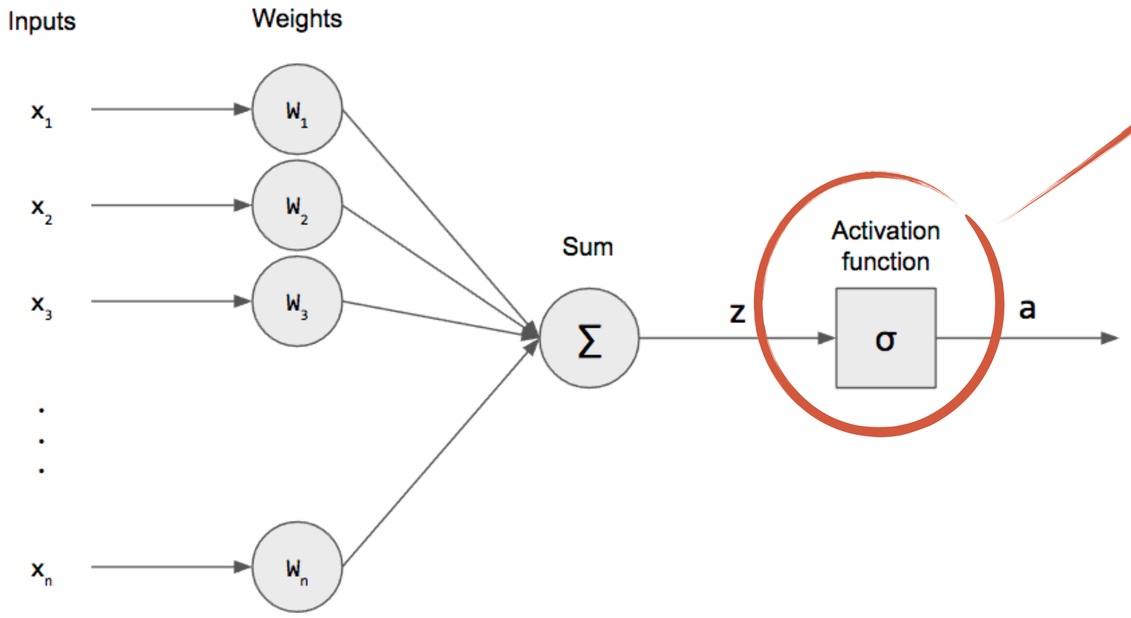
NEURON (PERCEPTRON)

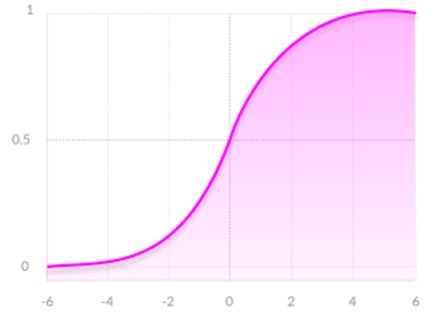
- It is the most basic unit in a neural network
- The weights [W_0 - W_n] is the relationship that we are looking for

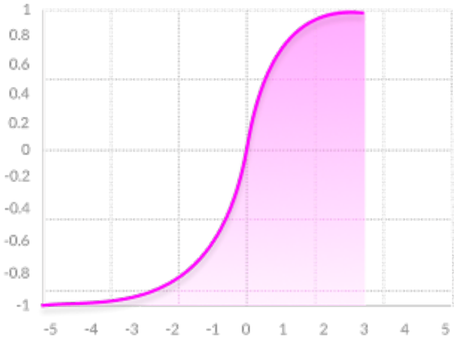


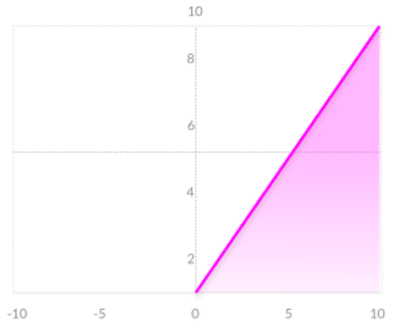
STEP FUNCTION AKA ACTIVATION FUNCTION

- A function to limit the output's data range
- Add non-linearity into a neural network









HOW TO INTERPRET NEURAL NETWORK RESULT (LOSS / ACCURACY)

- Loss is a measurement of the difference between the actual and predicted values
- Accuracy is a measurement of how many predictions the model has got right

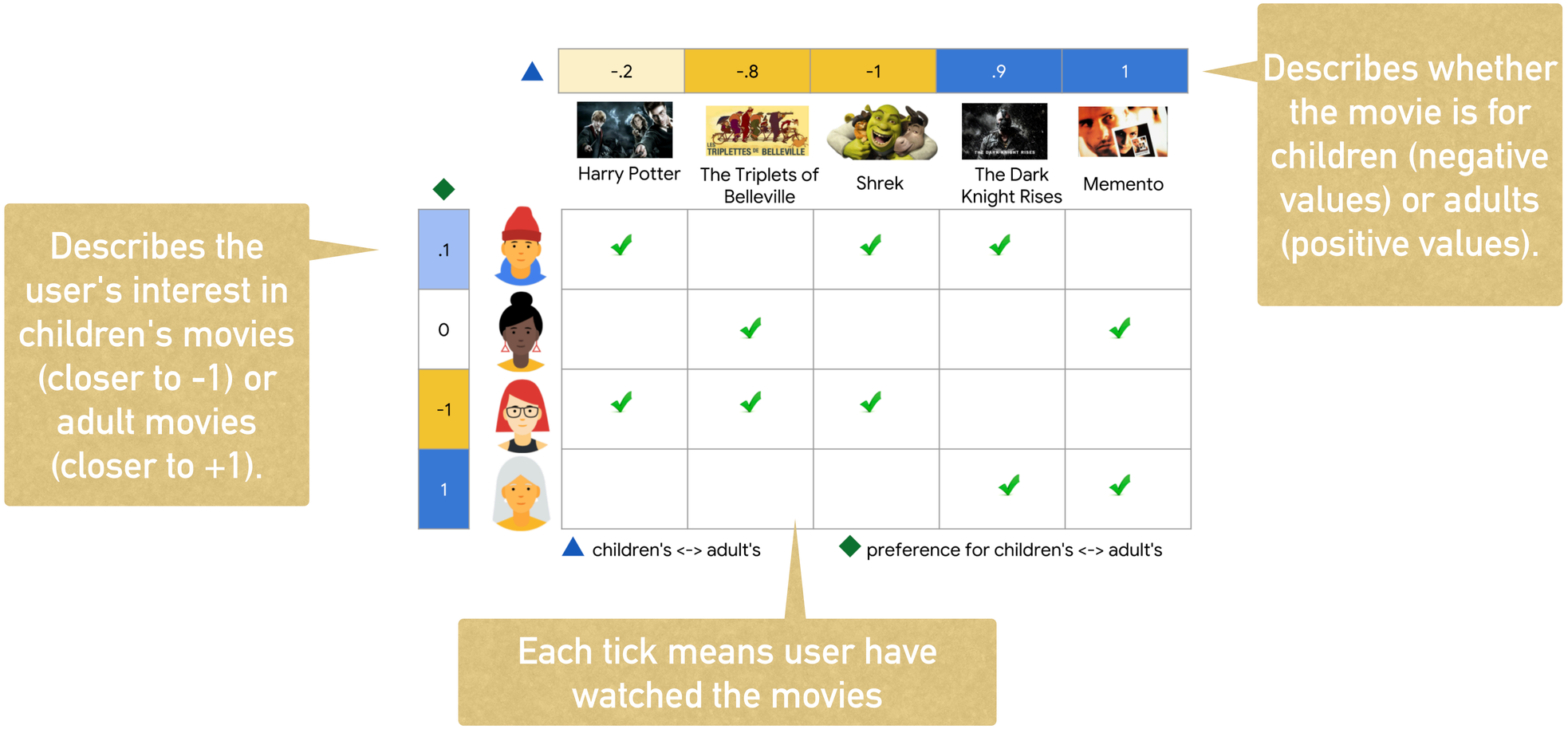
We train the neural network to workout 1 set of weights that work across the entire dataset (seen or unseen) and we are using the weight for predictions

HOW DO WE BUILD RECOMMENDER SERVICE (FROM PAST TO PRESENT)

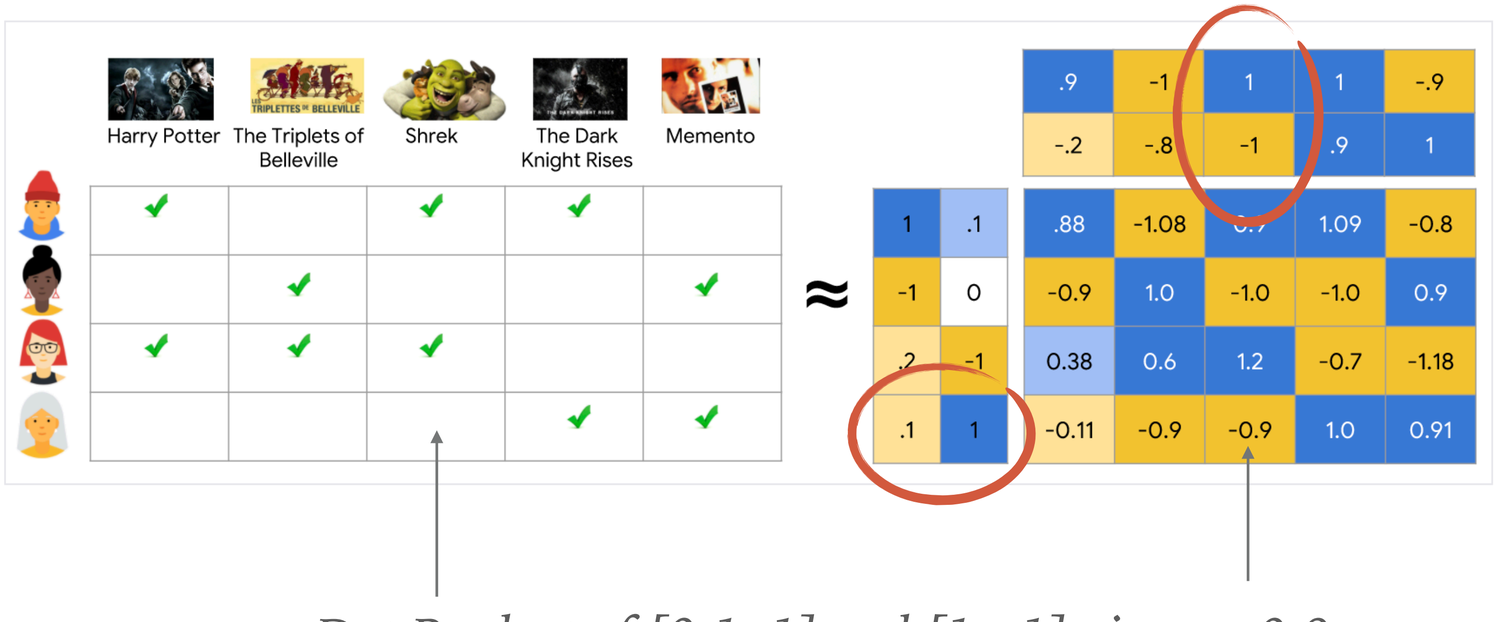
MATRIX FACTORISATION (MF) A VARIANT OF COLLABORATIVE FILTERING (CF)

WHAT IS A MATRIX FACTORIZATION (MF)

- A dot product of 2 matrix to approximate the level of interest between the customers and the products



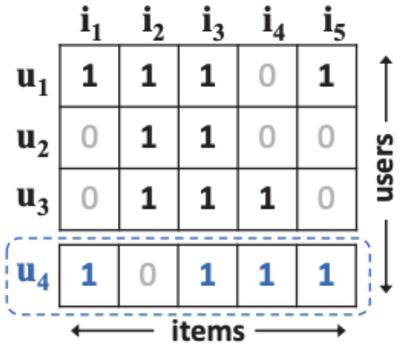

Describes whether the movie is Arthouse (negative values) or Blockbuster (positive values)
.9 -.8 1 1 -.9
-.2 -.8 -1 .9 1
Harry Potter
The Triplets of Belleville
Shrek
The Dark Knight Rises
Memento
Describes the user's interest in Arthouse (closer to -1) or Blockbuster (closer to +1)
1 .1
-1 0
.2 -1
.1 1
?
arthouse <-> blockbuster
preference for arthouse <-> blockbuster
children's <-> adult's
preference for children's <-> adult's
We can use dot product between 2 matrix to find the missing values



Dot Product of [0.1, 1] and [1, -1] give us -0.9

LIMITATION OF MATRIX FACTORIZATION (MF)

We can see very clearly that row U_4 is similar to row U_1, U_3 and then row U_2

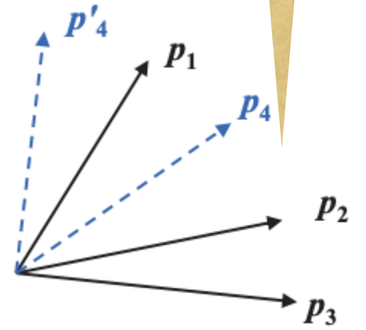


(a) user–item matrix

Transform

However if we transform each row to another plane, we see a different result

We can see that after transformation, P_4 is similar to P_1, P_2 then P_3 which contradict to the 1st conclusion, incurring a large ranking loss



(b) user latent space

LIMITATION OF MATRIX FACTORIZATION (MF)

- Cold Start
- if certain users have unusual interests (with respect to the rest of the users), it can cause errors in inference
- Hard to include side features for query / item
- Large ranking loss

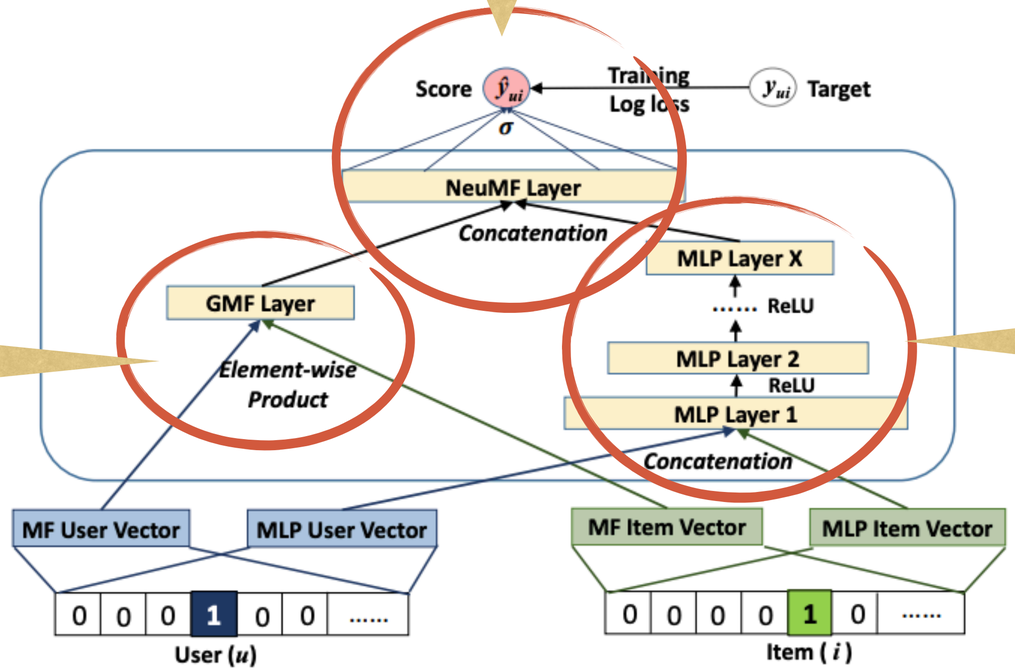
TO FIX THIS

USE OF NEURAL NETWORK IN COLLABORATIVE FILTERING (NCF)

NEURAL MATRIX FACTORISATION (NEUMF) A VARIANT OF NCF

5

Combine the Output



3

A Matrix Factorisation

4

Multi-Layer Perceptron

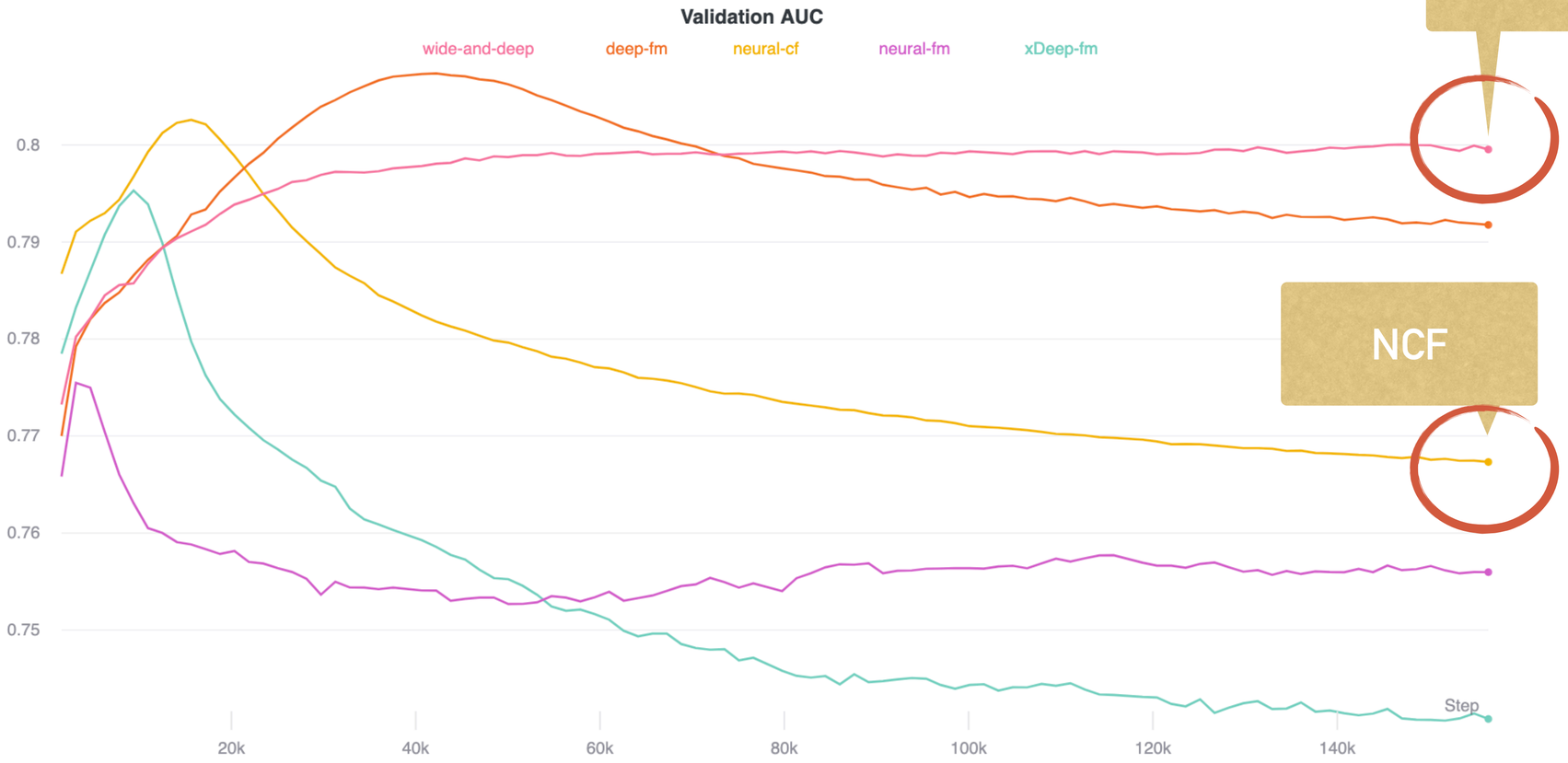
1

Each user(u) represent a user's data (age, occupation, country of origin etc)

2

Each Item (i) represent a product's data (name, risk, x years of return etc)

WHY NCF?



Wide-and-deep, developed by GOOGLE and launched on PlayStore (2006)

NCF

Model	Test AUC	Valid AUC	Runtime
wide-and-deep	0.7991	0.7995	1h12m15s
deep-fm	0.7915	0.7918	1h0m30s
xDeep-fm	0.7429	0.7408	2h15m17s
neural-fm	0.7589	0.7560	1h36m0s
neural-cf	0.7668	0.7673	54m15s

Minimal runtime with outstanding accuracy

On the Official TensorFlow

Mask R-CNN	Mask R-CNN
ShapeMask	ShapeMask: Learning to Segment Novel Objects by Refining Shape Priors
SpineNet	SpineNet: Learning Scale-Permuted Backbone for Recognition and Localization

Natural Language Processing

Model	Reference (Paper)
ALBERT (A Lite BERT)	ALBERT: A Lite BERT for Self-supervised Learning of Language Representations
BERT (Bidirectional Encoder Representations from Transformers)	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
NHNet (News Headline generation model)	Generating Representative Headlines for News Stories
Transformer	Attention Is All You Need
XLNet	XLNet: Generalized Autoregressive Pretraining for Language Understanding

Recommendation

Model	Reference (Paper)
NCF	Neural Collaborative Filtering

NCF is the only recommended model for Recommender service

How to get started with the official models

- The models in the master branch are developed using TensorFlow 2, and they target the TensorFlow nightly binaries built from the master branch of TensorFlow.
- The stable versions targeting releases of TensorFlow are available as tagged branches or downloadable releases.
- Model repository version numbers match the target TensorFlow release, such that release v2.2.0 are compatible with TensorFlow v2.2.0.

Please follow the below steps before running models in this repository.

Requirements

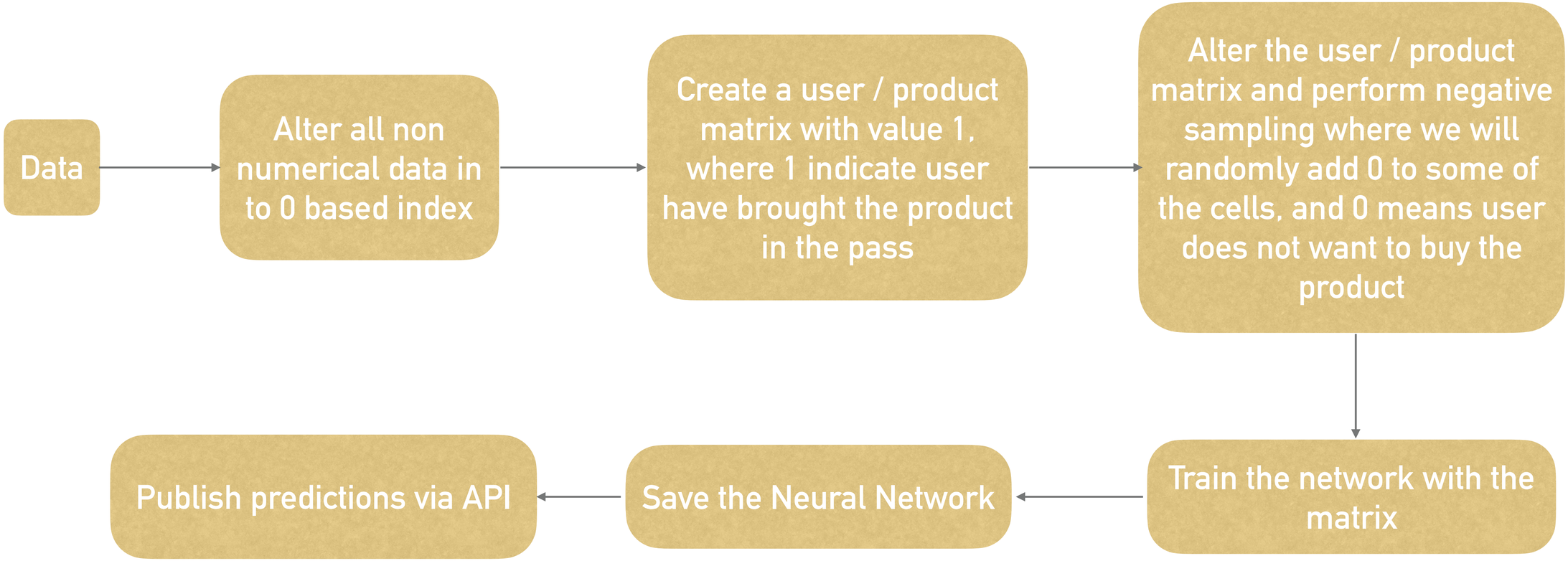
- The latest TensorFlow Model Garden release and TensorFlow 2
  - If you are on a version of TensorFlow earlier than 2.2, please upgrade your TensorFlow to the latest TensorFlow 2.

```
pip3 install tf-nightly
```

PROPOSITION

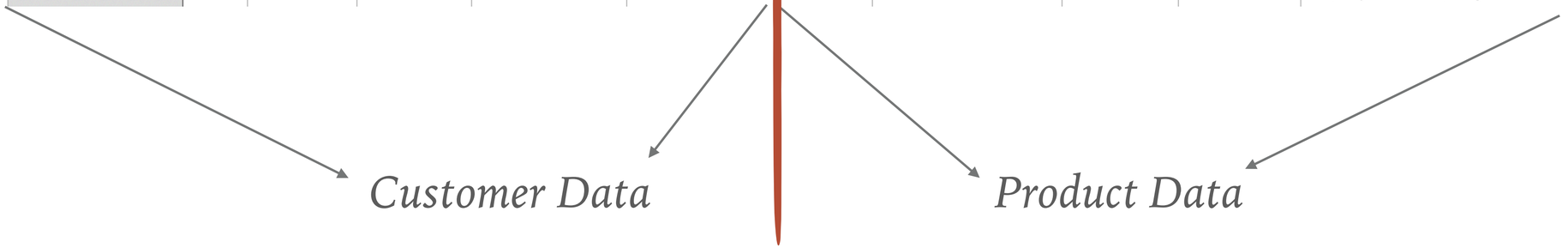
AIRS aim to compare different portfolios and predict whether a customer should be interested in certain products or not

BUILDING A RECOMMENDER SERVICE FROM SCRATCH



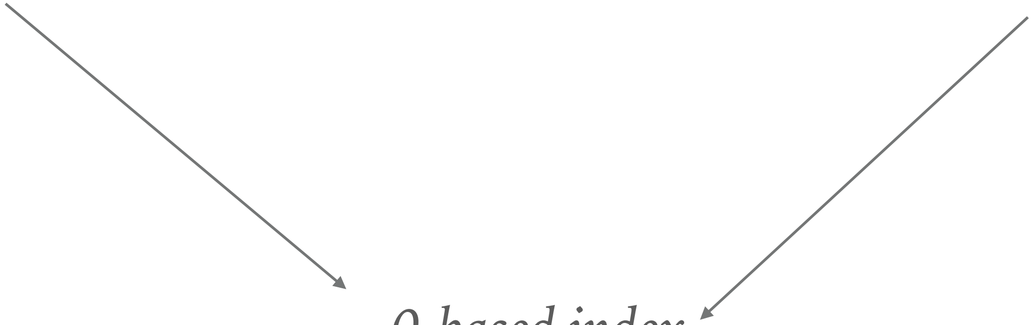
DATA

CUST_NUM	AGE	GENDER	MARITAL	HAVE_CHILD	EDU_LEVEL	CODE	3YEAR_RETURN	STD_DEV	DIVIDEND	ASSET_CLASS
CUST00000134	20	M	SINGLE	N	SECONDARY	U62300	11.37	20.12	0.22	Equity Developed Market
CUST00000155	20	M	SINGLE	N	SECONDARY	U61753	19.8	19.953	0	Equity Developing Market
CUST00000158	20	F	SINGLE	N	SECONDARY	U61753	19.8	19.953	0	Equity Developing Market
CUST00000201	20	M	SINGLE	N	SECONDARY	U61753	19.8	19.953	0	Equity Developing Market
CUST00000426	20	M	SINGLE	N	SECONDARY	U62402	5.9	24.169	4.9	Equity Developing Market
CUST00000537	20	M	SINGLE	N	SECONDARY	U61753	19.8	19.953	0	Equity Developing Market
CUST00000654	20	F	SINGLE	N	SECONDARY	U61753	19.8	19.953	0	Equity Developing Market
CUST00000655	20	F	SINGLE	N	SECONDARY	U61753	19.8	19.953	0	Equity Developing Market
CUST00000661	20	F	SINGLE	N	SECONDARY	U62402	5.9	24.169	4.9	Equity Developing Market



PRE-PROCESSING

user	age	gender	marital_status	have_child	education	product_name	3year_return	standard_deviation	dividend	asset_class	age_category	user_index	age_index	gender_index	education_index	have_child_index	marital_status_index	product_index	asset
CUST00000134	20	M	SINGLE	N	SECONDARY	U62300	11.37	20.12	0.22	Equity Developed Market	18-38	131	1	1	2	0	2	3	
CUST00000155	20	M	SINGLE	N	SECONDARY	U61753	19.8	19.953	0.0	Equity Developing Market	18-38	150	1	1	2	0	2	2	
CUST00000158	20	F	SINGLE	N	SECONDARY	U61753	19.8	19.953	0.0	Equity Developing Market	18-38	153	1	0	2	0	2	2	
CUST00000201	20	M	SINGLE	N	SECONDARY	U61753	19.8	19.953	0.0	Equity Developing Market	18-38	196	1	1	2	0	2	2	
CUST00000426	20	M	SINGLE	N	SECONDARY	U62402	5.9	24.169	4.9	Equity Developing Market	18-38	414	1	1	2	0	2	4	
CUST00000537	20	M	SINGLE	N	SECONDARY	U61753	19.8	19.953	0.0	Equity Developing Market	18-38	522	1	1	2	0	2	2	
CUST00000654	20	F	SINGLE	N	SECONDARY	U61753	19.8	19.953	0.0	Equity Developing Market	18-38	632	1	0	2	0	2	2	
CUST00000655	20	F	SINGLE	N	SECONDARY	U61753	19.8	19.953	0.0	Equity Developing Market	18-38	633	1	0	2	0	2	2	
CUST00000661	20	F	SINGLE	N	SECONDARY	U62402	5.9	24.169	4.9	Equity Developing Market	18-38	639	1	0	2	0	2	4	



USER / PRODUCT MATRIX

User/ Product	Product-A	Product-B	Product-C
User-1	1		
User-2	1		1
User-3		1	
User-4			1

NEGATIVE SAMPLING

User/ Product	Product-A	Product-B	Product-C
User-1	1	0	
User-2	1	0	1
User-3	0	1	
User-4		0	1

TRAINING

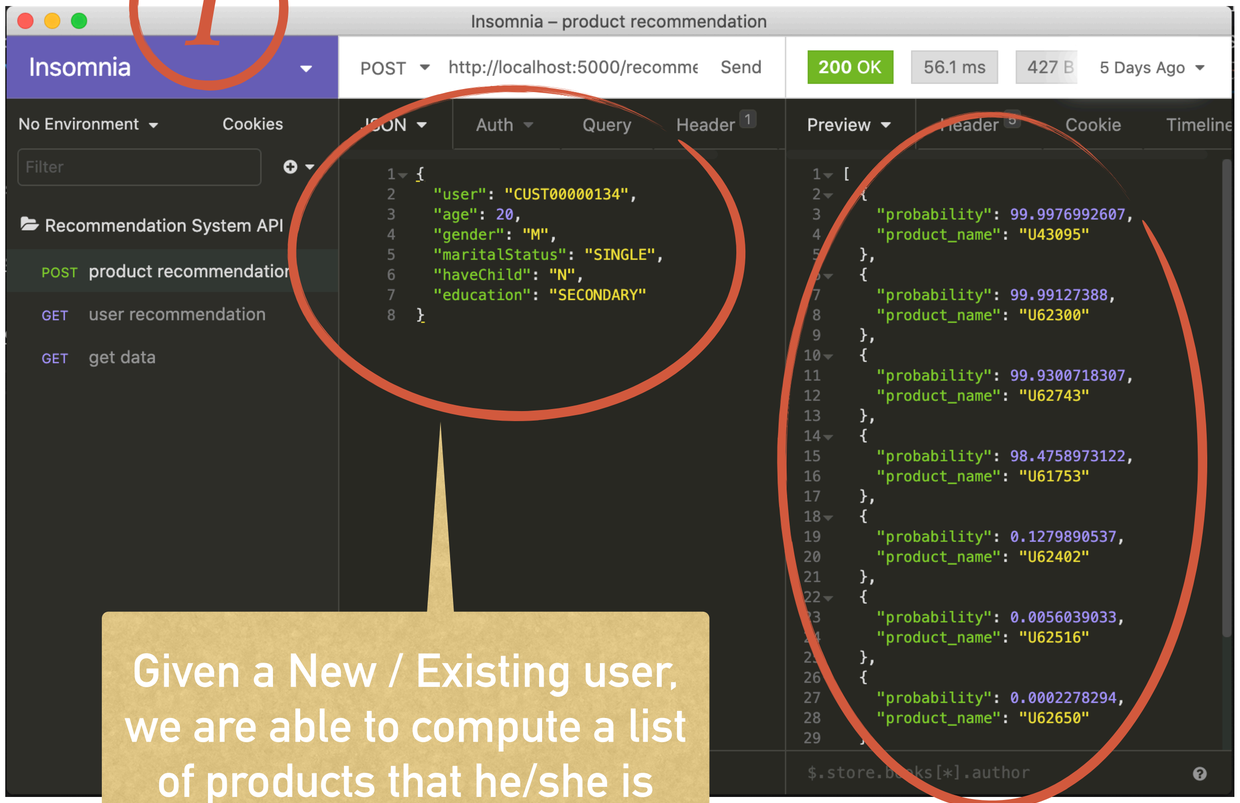
```
Number of devices: 1
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
user_input (InputLayer)         [(None, 6)]          0
__________________________________________________________________________________________________
item_input (InputLayer)         [(None, 6)]          0
__________________________________________________________________________________________________
mlp_user_embedding (Embedding)  (None, 6, 256)       248064      user_input[0][0]
__________________________________________________________________________________________________
mlp_item_embedding (Embedding)  (None, 6, 256)       6400        item_input[0][0]
__________________________________________________________________________________________________
flatten_2 (Flatten)             (None, 1536)         0           mlp_user_embedding[0][0]
__________________________________________________________________________________________________
flatten_3 (Flatten)             (None, 1536)         0           mlp_item_embedding[0][0]
__________________________________________________________________________________________________
concatenate (Concatenate)       (None, 3072)         0           flatten_2[0][0]
                                                                 flatten_3[0][0]
__________________________________________________________________________________________________
layer1 (Dense)                  (None, 256)          786688      concatenate[0][0]
__________________________________________________________________________________________________
layer2 (Dense)                  (None, 128)          32896       layer1[0][0]
__________________________________________________________________________________________________
layer3 (Dense)                  (None, 64)           8256        layer2[0][0]
__________________________________________________________________________________________________
mf_user_embedding (Embedding)   (None, 6, 64)        62016       user_input[0][0]
__________________________________________________________________________________________________
mf_item_embedding (Embedding)   (None, 6, 64)        1600        item_input[0][0]
__________________________________________________________________________________________________
layer4 (Dense)                  (None, 32)           2080        layer3[0][0]
__________________________________________________________________________________________________
flatten (Flatten)               (None, 384)          0           mf_user_embedding[0][0]
__________________________________________________________________________________________________
flatten_1 (Flatten)             (None, 384)          0           mf_item_embedding[0][0]
__________________________________________________________________________________________________
layer5 (Dense)                  (None, 16)           528         layer4[0][0]
__________________________________________________________________________________________________
multiply (Multiply)             (None, 384)          0           flatten[0][0]
                                                                 flatten_1[0][0]
__________________________________________________________________________________________________
layer6 (Dense)                  (None, 8)            136         layer5[0][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 392)          0           multiply[0][0]
                                                                 layer6[0][0]
__________________________________________________________________________________________________
result (Dense)                  (None, 1)            393         concatenate_1[0][0]
==================================================================================================
Total params: 1,149,057
Trainable params: 1,149,057
Non-trainable params: 0
__________________________________________________________________________________________________
```

```
Epoch 21/40
11/11 [==============================] - 0s 17ms/step - loss: 0.6005 - accuracy: 0.9015
Epoch 22/40
11/11 [==============================] - 0s 17ms/step - loss: 0.5942 - accuracy: 0.9198
Epoch 23/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5930 - accuracy: 0.9101
Epoch 24/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5906 - accuracy: 0.9327
Epoch 25/40
11/11 [==============================] - 0s 18ms/step - loss: 0.5866 - accuracy: 0.9316
Epoch 26/40
11/11 [==============================] - 0s 17ms/step - loss: 0.5820 - accuracy: 0.9408
Epoch 27/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5785 - accuracy: 0.9464
Epoch 28/40
11/11 [==============================] - 0s 21ms/step - loss: 0.5790 - accuracy: 0.9606
Epoch 29/40
11/11 [==============================] - 0s 18ms/step - loss: 0.5844 - accuracy: 0.9505
Epoch 30/40
11/11 [==============================] - 0s 18ms/step - loss: 0.5749 - accuracy: 0.9613
Epoch 31/40
11/11 [==============================] - 0s 18ms/step - loss: 0.5769 - accuracy: 0.9628
Epoch 32/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5729 - accuracy: 0.9651
Epoch 33/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5716 - accuracy: 0.9640
Epoch 34/40
11/11 [==============================] - 0s 18ms/step - loss: 0.5741 - accuracy: 0.9668
Epoch 35/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5699 - accuracy: 0.9655
Epoch 36/40
11/11 [==============================] - 0s 18ms/step - loss: 0.5754 - accuracy: 0.9623
Epoch 37/40
11/11 [==============================] - 0s 18ms/step - loss: 0.5728 - accuracy: 0.9677
Epoch 38/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5718 - accuracy: 0.9657
Epoch 39/40
11/11 [==============================] - 0s 19ms/step - loss: 0.5671 - accuracy: 0.9684
Epoch 40/40
11/11 [==============================] - 0s 20ms/step - loss: 0.5709 - accuracy: 0.9662
```

AIRS-API

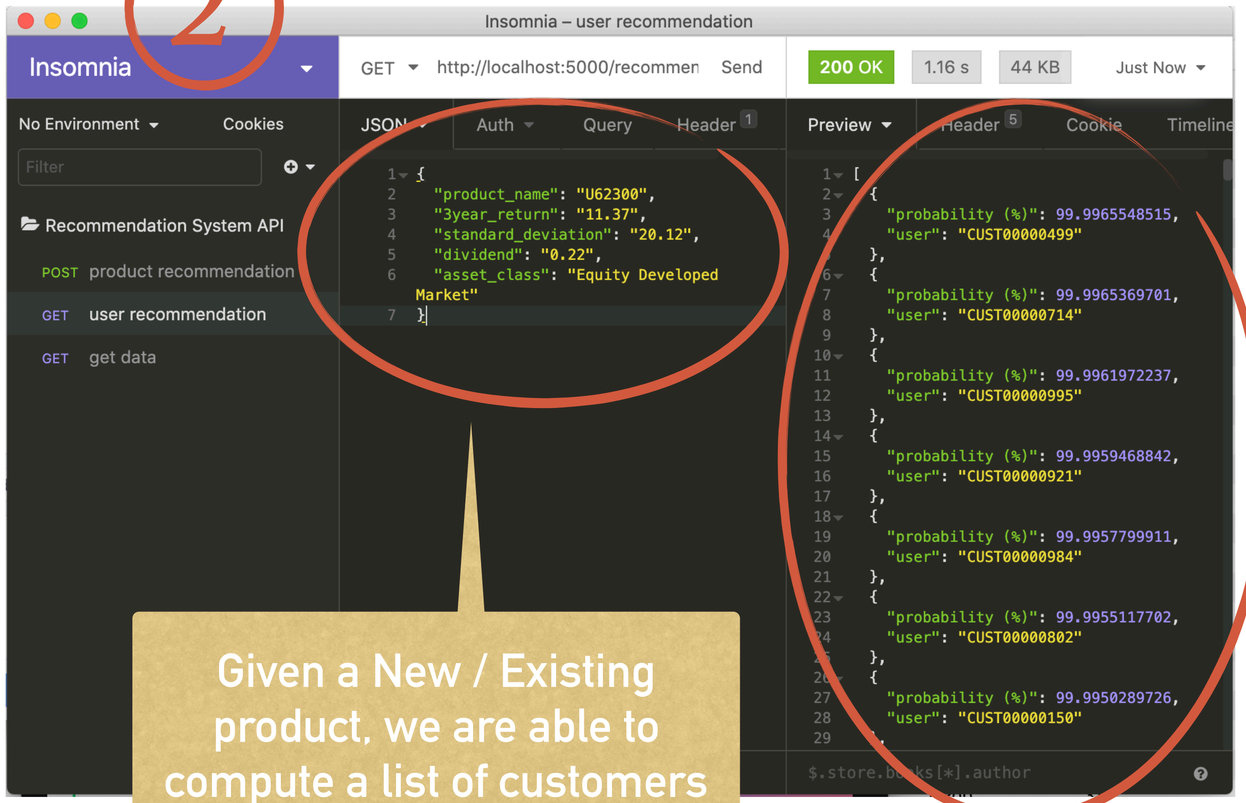
https://github.com/OneDodge/financial-product-recommendation-system/

1



Given a New / Existing user, we are able to compute a list of products that he/she is interested in

2



Given a New / Existing product, we are able to compute a list of customers that they are interested in

DEMO

IMPROVEMENTS FOR THE FUTURE

- labeling data with 1 and 0 as what we want to predict yield a relatively high loss in training the neural network
- due to a limited dataset available, left us very little room for maneuver
- We require rich data set from various sources i.e.
  - How long did a user use the app, web, length of the meeting etc
  - How long did a user browse a product
- The model currently did not account for time series pattern, i.e.
  - If there is an economic breakdown, holding stocks might not be a good choice, in this case, the model should return gold, or bonds instead.
- The model is unable to handle sudden change in the market

QUESTION