Recommender System

Most Valuable application of ML- Recommender System

Recommender System

Daily Life Activities
-what songs you might like to listen,
-what food to order online,
-what posts you see on your favorite social networks,
-next person you may connect with,
-what series or movies you would like to watch, etc....

Netflix

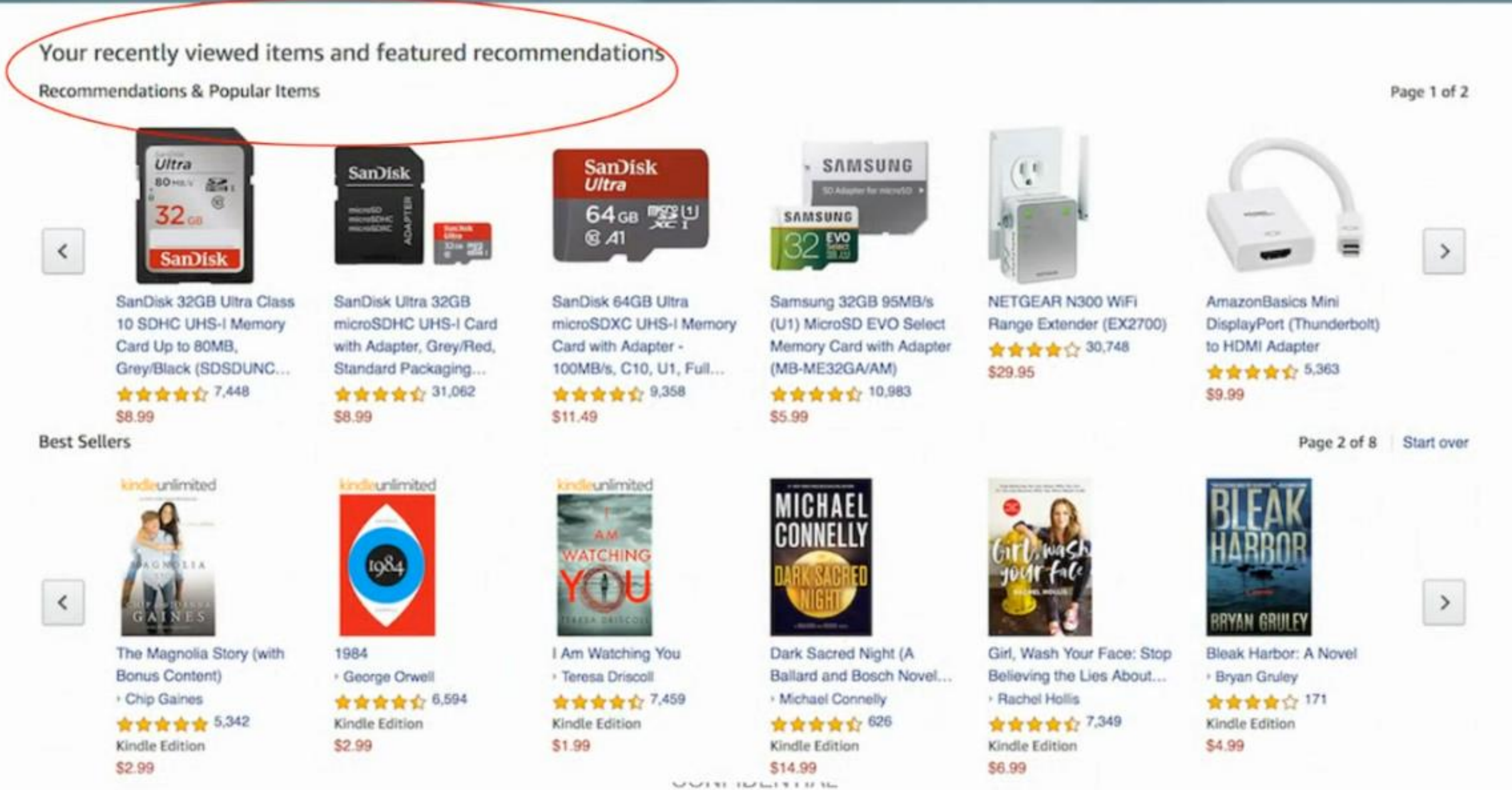
Recommender System

Over 80% of what members watch comes from our recommendations

Recommendations are driven by Machine Learning Algorithms

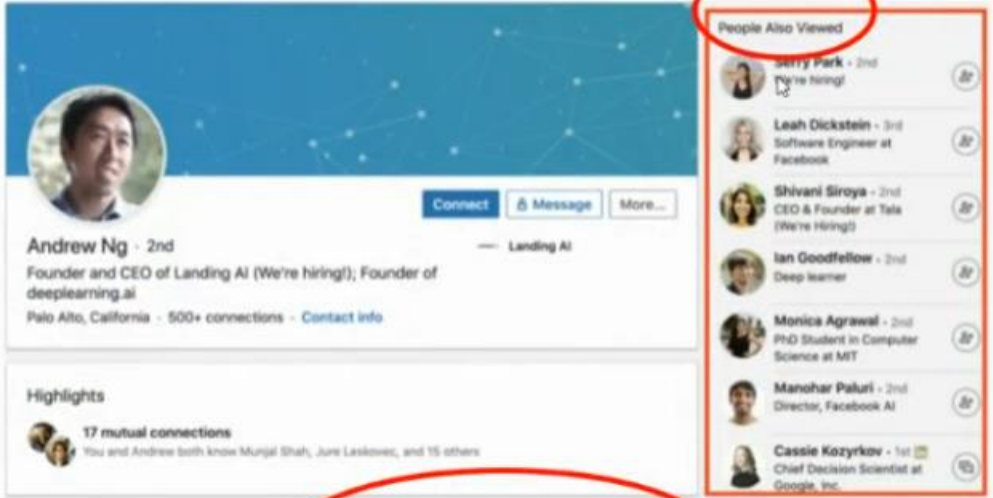
35% of what customers purchase at Amazon comes from product recommendations

Recommender System

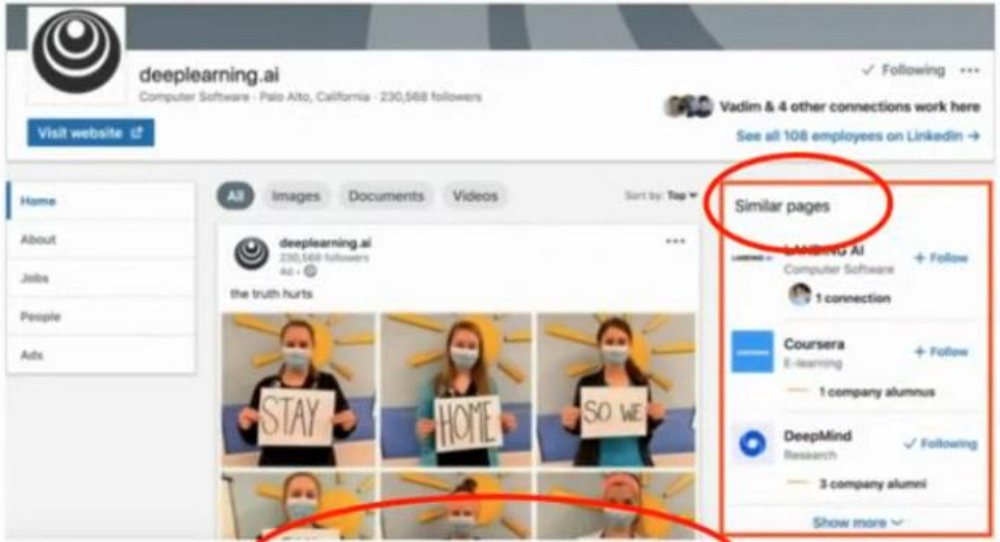


Recommendations on LinkedIn

Recommender System



Similar User Recommendations on Linkedin



Similar Companies Recommendations on Linkedin

Most companies want to recommend the most relevant content to their users which they enjoy in the future

Recommender System

- **Similar** home listings (Airbnb, Zillow)
- **Relevant** media, e.g., photos, videos and stories (Instagram)
- **Relevant** series & Movies (Netflix, Amazon Prime Video, Hot star)
- **Relevant** songs and podcasts (Spotify)
- **Relevant** Videos (YouTube)
- **Relevant** dishes & restaurants (Zomato, Uber Eats)
- **Similar** Users, Posts (LinkedIn, Twitter, Instagram)

The Classic approach to recommender system is based on Collaborative Filtering (Used by Amazon, Netflix, LinkedIn, Spotify and You Tube)

Input Data

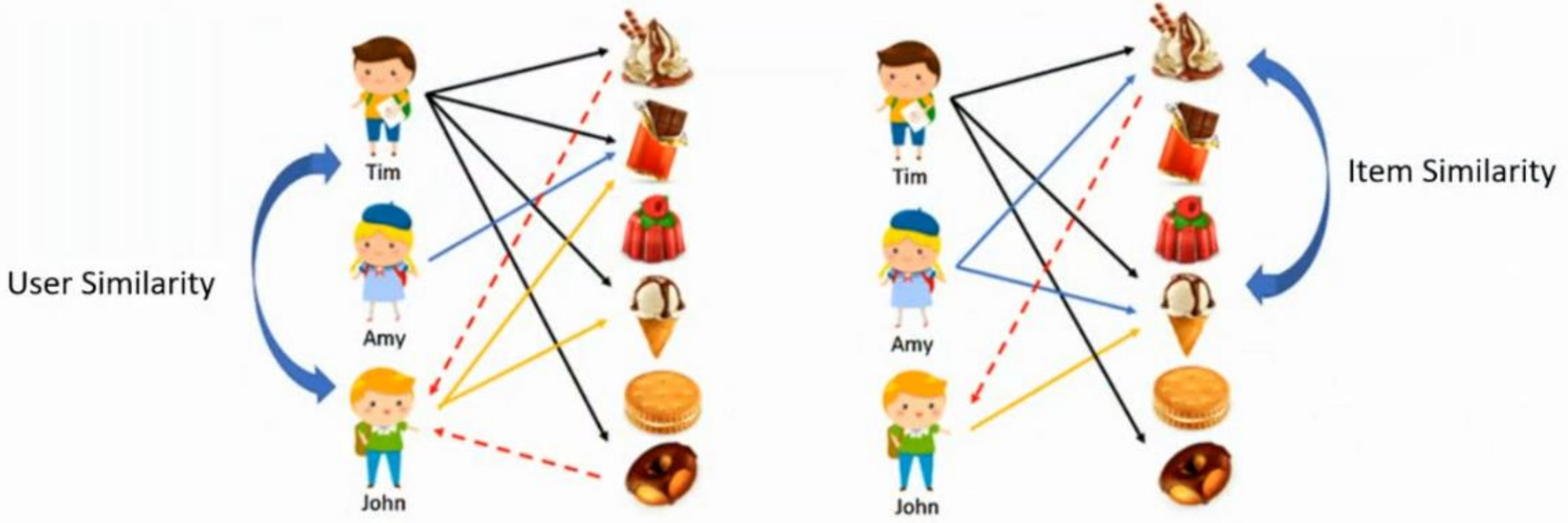
Recommender System

- **User Interaction Data**
- Consists of **User - User or Item - Item relationships** to find similar content.
- Each User will have features, similarly each item will have features.
- This is the data we gather from the web logs.
- This data can be divided into two groups:
  - **Explicit Data**--> Explicit Input from Users (e.g., Movie ratings, Search logs, liked, commented, watched , favorited etc.)
  - **Implicit Data**--> Information that is not provided intentionally but gathered from available data streams (e.g., search history, order history, clicked on, accounts interacted with etc.)

- **Item Data (Features)**
- Consists mainly of an item's features.
  - For You Tube it would be a video's metadata such as title description, theme, duration, cast etc.
  - For Airbnb it could be a home's postal code, city region, no of bedrooms etc.
- Other Data sources could be **External Data**
  - For example, Netflix might add external item features like box office performance or critic reviews

Candidate Generation → Collaborative Filtering

Recommender System



(a) User-based filtering

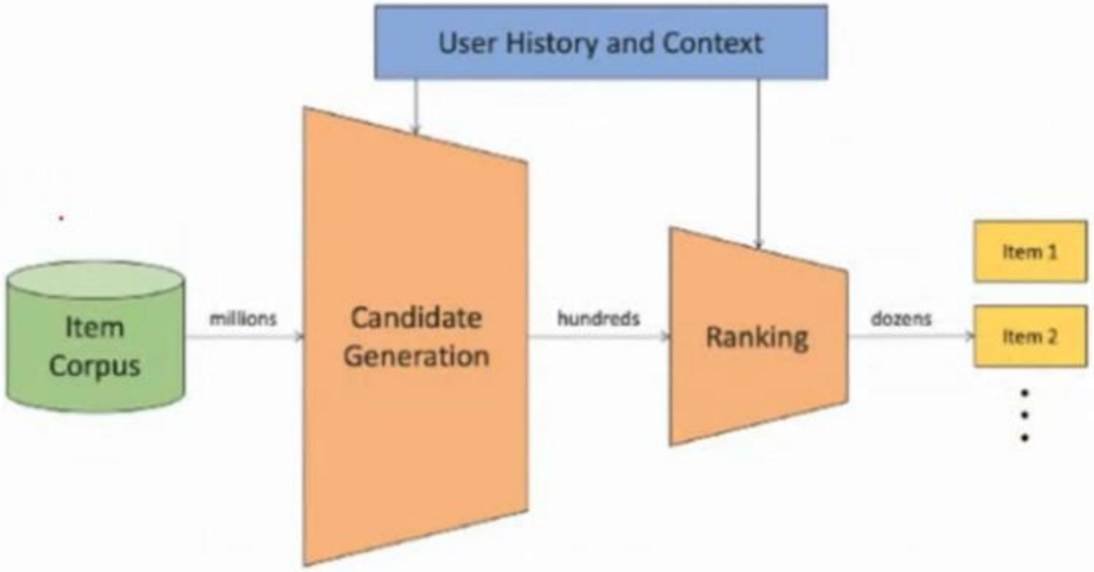
(b) Item-based filtering

- **Tim is Similar to John (as likings are more in comparison to Amy)**
- **So, John is recommended items which are in the list of Tim but are not in the list of John.**

- **Ice-creams are Similar**
- **So, John is recommended items which are in the list of Tim but are not in the list of John.**

Collaborative Filtering (Two Stage Recommender System)

Recommender System



2-stage Recommender System (inspired by YouTube)

Two Stages:

-**Candidate Generation**--> Sourcing the relevant candidates that could be eligible to show to our users.

-**Ranking** → Sorting the above list in the order of expected enjoyment (utility)

For Example--> In case of Netflix, the Goal is to present several attractive items for a person to choose from

a**) Item Corpus** --> Movies, Serials, Documentaries etc..

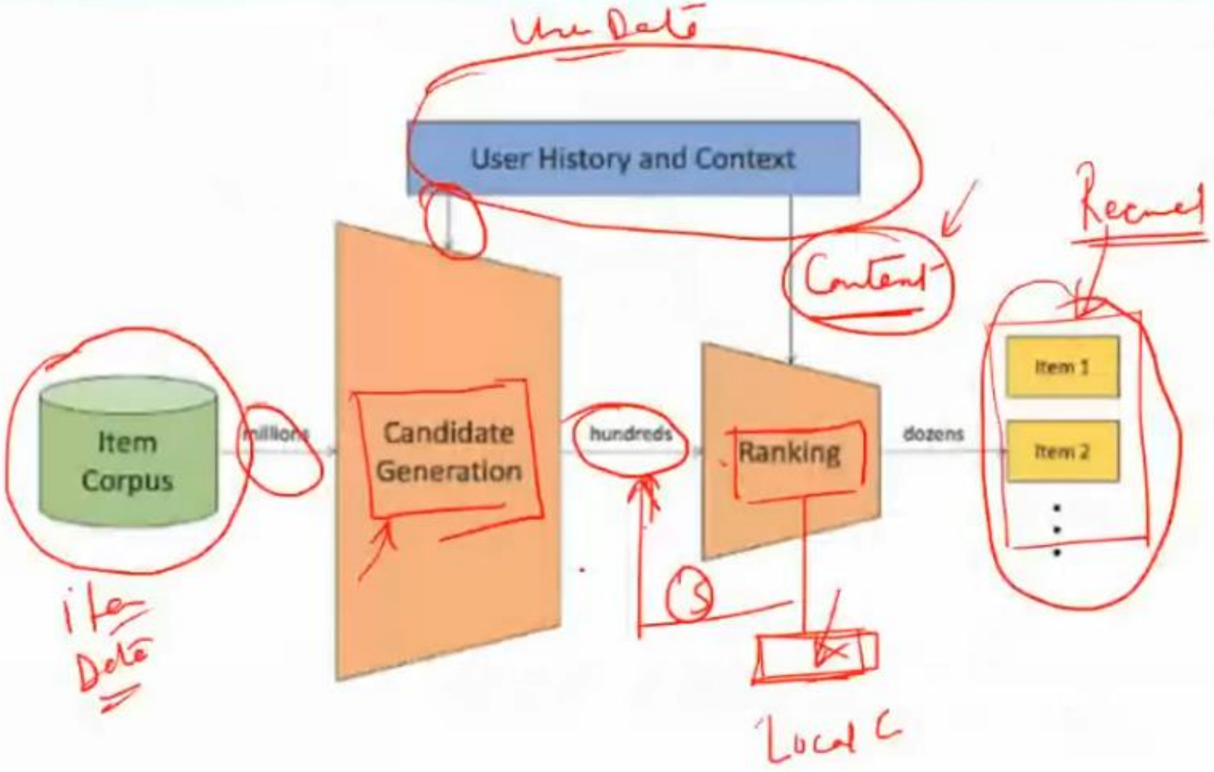
b) **User History & Contex**t--> Who is watching (Context), What watching, Search logs (User History)

Collaborative Filtering → Generating User Vector & Item (Movies) Vector

Recommender System

- In case of the Netflix, We generate User Vector & Item (Movies) Vector (Also Called Embeddings)
- **User Vector:** To find out which users' tastes are similar-to another's, collaborative filtering compares one user's vector with all the other user's vectors (distance can be calculated by Euclidian Distance Formula), ultimately popping out which users are the closest matches. (Distance between Similar vectors will be less in comparison to dissimilar vectors)
- **Item (Movies) Vector:** The same goes for Movie's vector, we can compare each movie vector with all the others and find out which movies are most similar-to the one in question.
- These Video Embeddings are learned by the Algorithm.

Collaborative Filtering (Two Stage Recommender System)



2-stage Recommender System (inspired by YouTube)

Two Stages:

-**Candidate Generation**--> Sourcing the relevant candidates that could be eligible to show to our users.

-**Ranking →** Sorting the above list in the order of expected enjoyment (utility)

For Example--> In case of Netflix, the Goal is to present several attractive items for a person to choose from

a) **Item Corpus** --> Movies, Serials, Documentaries etc..

b) **User History & Context**--> Who is watching (Context), What watching, Search logs (User History)

Collaborative Filtering → Generating User Vector & Item (Movies) Vector

Recommender System

- Once we have vectorial representation of our users & items, we can simply use K nearest neighbor search to find our potential candidates.
- By doing this we can filter billions of media items down to couple of thousand and then sample 500 candidates from the pool and send those candidates downstream to the ranking stage.
- This phase is guided by user Input (Say Movies --> Action, Horror Romantic….watched, liked, commented, favorited by user)
- Apart from Collaborative Filtering Other Algorithms can also be used for example
  - for word embeddings--> word2vec in NLP
  - for Dish & Restaurant embeddings --> Graph Learning
  - Neural Networks

User Vector & Item Vector

$$\text{Users} \begin{pmatrix} 1&0&0&0&1&0&0&1 \\ 0&0&1&0&0&1&0&0 \\ 1&0&1&0&0&0&1&1 \\ 0&1&0&0&0&1&0&0 \\ 0&0&1&0&0&1&0&0 \\ 1&0&0&0&1&0&0&1 \end{pmatrix}$$

Movie / Songs

Here User ‘X’ has watched a Movie ‘Y’ or listened to Song ‘Y’)

Stage 2: Ranking

- For Ranking ML algorithms which learn to rank are used.
- In the ranking stage, we are not aiming for our items to have a global notion of relevance, but rather looks for ways of optimizing a personalized model (Based on users' contextual information)
- For example: In case of a Zomato, it will be time of day, current location of user when they open Zomato app).
- Many supervised learning methods can be used for ranking --> Logistic regression, Support Vector machine, neural networks, decision-tree based methods or specifically designed algorithms such as Rank SVM & Rank Boost

Evaluating Recommender System Metrics

Recommender System

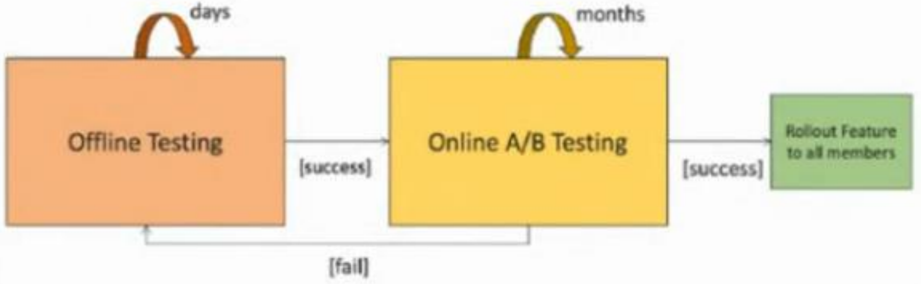
Evaluation Metrics --> Two Types

Offline --> Low level metrics, e.g., performance of model per se (RMSE). We need to pick the right metrics for each stage. (candidate generation & ranking)

- Candidate generation will focus on high precision "*out of all the videos that were preselected how many are relevant'*
- In the ranking stage, goal is to present a few 'best' recommendations this will focus on high recall ' *how many of the relevant videos did we find'*

Online --> High level business metrics that are measurable as soon as we ship our model into real world and test it with real users.

- Some examples include A/B testing, customer retention, click thru rate, user engagement etc.



Offline/Online Testing Framework

A/B testing is a strategy that pits two different versions of a website, advertisement, email, popup, against each other to see which is most effective.

Cold Start Problem

Recommender System

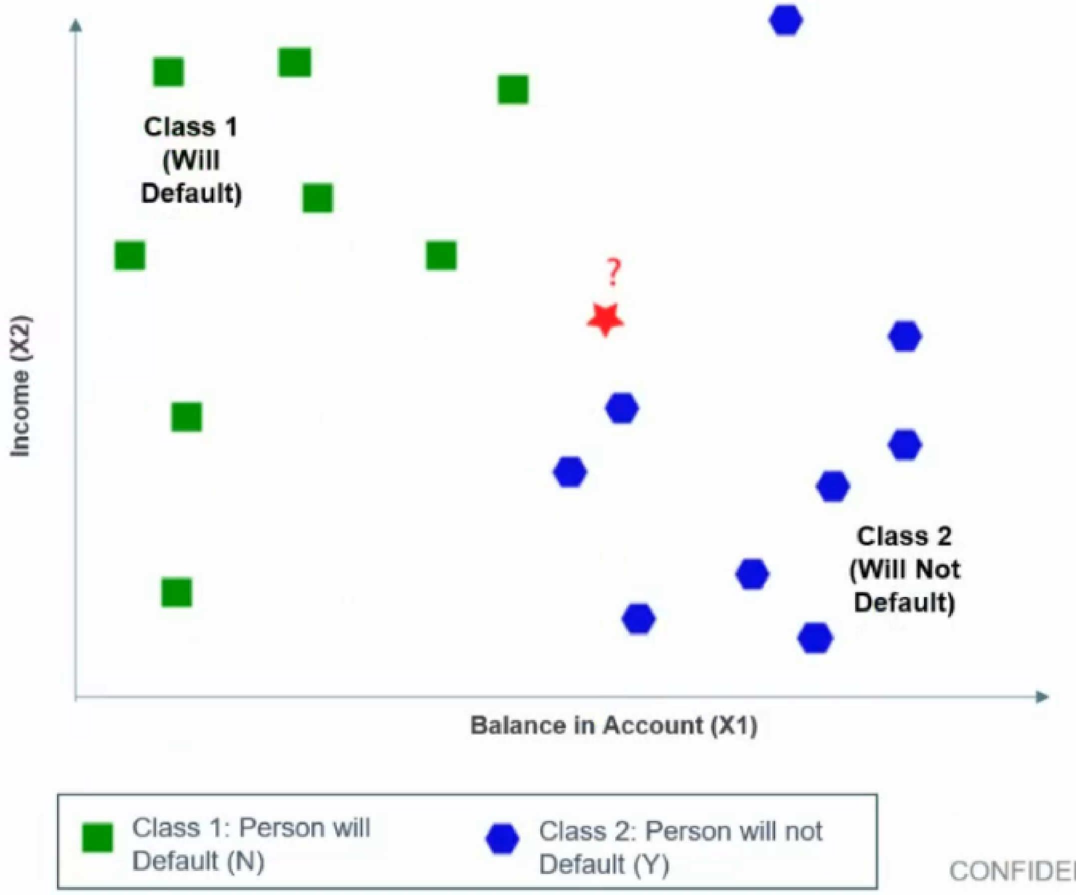
- Methods such as collaborative filtering rely heavily on past user item interactions giving rise to Cold Start problem. This is of two types:

- **User Cold Start** --> Imagine a new member signs up for Netflix, At this point, the company does not know anything about new members preferences. How does the service keep the new member engaged by providing great recommendation?
  - One way it is being handled is new members (as a part of sign-up) process are asked to select videos from an algorithmically populated set that is then used as an input into all of their algorithms.

- **Item Cold Start** --> Similar Challenge as above, when new item (say Movie is added)
  - This is achieved by locating 3 closest listings that have similar embeddings (such as genre, price etc.) as that of new content and calculate their mean vector and assign to the new content

Taking K=3 & K=5….

KNN



K Value	Blue Dots	Green Dots	Class Assigned to Red Star
3	2	1	Blue
5	2	3	Green

Evaluating Recommender System Metrics

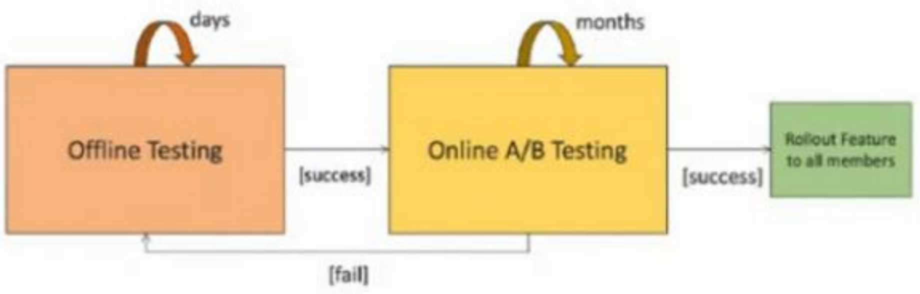
Evaluation Metrics --> Two Types

Offline --> Low level metrics, e.g., performance of model per se (RMSE). We need to pick the right metrics for each stage. (candidate generation & ranking)

- Candidate generation will focus on high precision "*out of all the videos that were preselected how many are relevant'*
- In the ranking stage, goal is to present a few 'best' recommendations this will focus on high recall *' how many of the relevant videos did we find'*

Online --> High level business metrics that are measurable as soon as we ship our model into real world and test it with real users.

- Some examples include A/B testing, customer retention, click thru rate, user engagement etc.



Offline/Online Testing Framework

A/B testing is a strategy that pits two different versions of a website, advertisement, email, popup, against each other to see which is most effective.