Deep Dive Series: <https://www.youtube.com/playlist?list=PLhr1KZpdzukd9GSGRy329wahNO_8TkRo_>

<https://towardsdatascience.com/build-a-recommender-system-in-less-than-an-hour-using-amazon-personalize-68bee9931c60>

Cheatsheet: <https://github.com/aws-samples/amazon-personalize-samples/blob/master/PersonalizeCheatSheet2.0.md>

**What are ARNs** : Amazon Resource Names (ARNs) **uniquely identify AWS resources**. We require an ARN when you need to specify a resource unambiguously across all of AWS

**IAM role** : AWS Identity and Access Management (IAM) roles **provide a way to access AWS by relying on temporary security credentials**. Each role has a set of permissions for making AWS service requests, and a role is not associated with a specific user or group.

**S3 Policy** : A bucket policy is **a resource-based AWS Identity and Access Management (IAM) policy**. You add a bucket policy to a bucket to grant other AWS accounts or IAM users access permissions for the bucket and the objects in it.

**Dataset Group** is the largest isolated container of information inside Amazon Personalize. So any information inside one dataset group can’t adverse or really impact other dataset groups, either within your own account or across accounts.This makes them ideal for isolating experiments, and finding which recommendation system works best.

Recipes:

1. User Personalization : Given a user, recommend items based on interactions by similar users, uses user-user collaborative filtering
2. Similar-Items : Recommends similar items wrt a given item, based on item-metadata. Finds similarity between data.
3. SIMS : Recommends items wrt a given item, uses item-item collaborative filtering, by looking at how people are co-interacting with items and not metadata. Use Case: Recommend similar items in store based on similar buying history. Food ordering system, recommend similar kinds of foods depending on which food items were bought together.
4. Personalized Ranking : Looks at the collection of items and ranks them in order of most relevant to least for the given user.
5. Popularity Count : Not machine learning, just a baseline from counting the most commonly interacted items. Gives “Most popular”, “Most viewed”, and “Best sellers” kind of data.
6. User segmentation : Create segments of users based on their affinity to specific items in your catalog or their affinity to item attributes.

Sample dataset to understand these recipes:

Users : Vijay, Rayster, Prajyot, Piyush, Adit, Amit, Ayush

ItemIds : 1, 2, 3, 4, 5

User-Item Interaction Datatset:

| User | ItemId | Timestamp |
| --- | --- | --- |
| Vijay | 1 | 1 |
| Adit | 2 | 1 |
| Vijay | 5 | 2 |
| Prajyot | 3 | 2 |
| Amit | 4 | 3 |
| Ayush | 2 | 3 |
| Rayster | 1 | 3 |
| Piyush | 5 | 3 |
| Prajyot | 2 | 4 |
| Rayster | 3 | 4 |
| Ayush | 5 | 4 |
| Piyush | 1 | 4 |
| Adit | 5 | 4 |
| Vijay | 2 | 5 |

Item Dataset:

| ItemId | Metadata | Timestamp |
| --- | --- | --- |
| 1 | Spices, Food ingredient | 1 |
| 2 | Soap, Personal Hygiene | 2 |
| 3 | Cooking Oil, Food ingredient | 2 |
| 4 | Conditioner, Personal Hygeine | 3 |
| 5 | Shampoo, Personal Hygiene | 3 |

1. User Personalization: Selected User: Piyush, items bought by user are 1, 5. Find users who bought these : Adit, Ayush, Rayster, Vijay and recommend their bought items to Piyush. This also takes into account the timestamp i.e. prefers the most recent interactions.
2. Similar Items : Selected Item : 1, Recently bought items with this item by a given user : [(5(by Piyush)), (3(by Rayster)), (5,2(by Vijay)]. By looking into these items and most recent interactions, items can be recommended.
3. SIMS : Selected item : 4, similar items based on its metadata are 2, 5(Personal Hygiene)
4. Personalized Ranking : Selected user : Rayster, items : 2, 4, 5 looking at his interaction history, rate these items
5. User Segmentation : Selected item : 3, create a segment of people buying these items [Prajyot, Rayster] . Selected item attribute : Food ingredient, it will segment all the users who are buying items with metadata ‘Food ingredient’ : [Vijay, Prajyot, Rayster, Piyush]
6. Popularity Count:

1 : 3

2 : 4

3 : 2

4 : 1

5 : 4

Using this count we can tell most popular items are : [2, 5]

**Solution:** Combination of an Amazon Personalize recipe and customized hyperparameters that you select.

**Solution version:** Trained Machine Learning model that you can deploy to get recommendations for customers, parameters are defined via a solution.

Read more on recipes

Solutions and their Solution Versions are bound to a particular dataset group

**Metrics:**

1. Coverage: Proportion of items that are being recommended from total dataset.

**HyperParameter Optimization:**

1. min/max\_user\_history\_length\_percentile(All): Exclude from training a percentage of users based on short or long histories.
2. exploration\_weight(User Personalization): Determines how frequently recommendations try new, something experimental.
3. exploration\_item\_age\_cutoff(User Personalization): Determines to be explored based on timeframe since latest interaction.
4. popularity\_discount\_factor(SIMS): Affects balance between popularity and correlation(Reduce popular items)
5. min\_correlation\_count(SIMS): Sets the limit on minimum cointeractions needed to compute similarity score for a pair of items.

**EventTracker:** An event-tracker provides an endpoint that allows you to stream interactions that occur in your application back to personalize.This is done by PutEvents API.

**Minimum Provisioned Transactions per second:** Tells personalize that this number is my baseline or my low watermark for request volume that I expect to send to this particular campaign. Set this value as low as possible. Setting it lower will tell personalize to auto scale things up. So personalize will scale things up as volume increases and then scale down.

We are charged based on max of either actual volume or minimum provisioned TPS, we have to make sure it is as low as possible so we are not paying for resources we are not using.

**Batch Inference Job:** Create an input file in JSON format with given UserIds or ItemIds, personalize will return an output file also in JSON format with recommendations to our S3 bucket.

How Amazon Personalize work:

1. Create and load dataset:
   1. Prepare a CSV file with who bought what data - interactions dataset
   2. Add this CSV file to a S3 bucket.
   3. Launch Amazon Personalize
   4. Go to Create Dataset Group
   5. In overview, click on Import User-Item interaction dataset
   6. Here, we must create a dataset name and provide JSON Schema for the dataset. A sample JSON schema:

schema **=** {

"type": "record",

"name": "Interactions",

"namespace": "com.amazonaws.personalize.schema",

"fields": [

{

"name": "USER\_ID",

"type": "string"

},

{

"name": "ITEM\_ID",

"type": "string"

},

{

"name": "TIMESTAMP",

"type": "long"

}

],

"version": "1.0"

}

* 1. After this, start import by providing the S3 location of data.

1. Creating a Solution: