SE PROJECT RESEARCH

1. Basic Idea of Buyer Persona: https://www.shopify.in/retail/know-your-customers-how-to-build-buyer-personas-for-your-retail-store

Step 1: Identify group of buyers
To start off defining your buyer personas, you'll need to identify the broad groups of consumers that you have. Use your purchase data for it. The best data for developing your buyer personas is what you know about your end customers.

Step 2: Identify Key information
After defining your customers into broad categories, you'll need to identify what more granular information you'll need to create the personas like Location, Age, Gender, Interests, etc.

Step 3: Gather Information
Now that you've identified the information that you'll need to build your buyer personas, you'll need to start gathering data and intel. Start with the information you already have about your customers. Scrolling through customer data as well as their order history can be enlightening when crafting your buyer personas.

Requested data from Shopify. No reply yet.

2. Clustering for Buyer Persona: https://www.researchgate.net/publication/236736903_Using_cluster_analysis_in_Persona_development

Goodwin defines Personas as: "User models, or Personas, are fictional, detailed archetypical characters that represent distinct groups of behaviours, goals and motivations observed and identified during the research phase."

A Persona is created by analysing the real users' goals, behaviours and motivations. The real users' data may come from marketing research, user study which includes interviews, questionnaires, observations, and ethnographic studies, etc.

Marketing data and ethnography studies data may consist of large quantity of multidimensional data sets. The statistical method Principal Components Analysis (PCA) can be used to reduce the dimensionality of the data while Cluster Analysis (CA) can group the data based on their similarities.

Clustering of users based on personal information:
1. Collect basic info through registration process like age, gender.
2. Online form or questionnaire to collect more info like cuisine, common allergies, taste buds, lifestyle, Veg/Non-veg/Vegan, etc.
3. Matrix of User vs User_Info_Attributes collected from above process. Here, Each user is encoded in terms of attributes like age, gender, cuisine, etc.
4. Clustering algorithm used: Hierarchical clustering with complete linkage and euclidean distance as similarity metric.
5. Hierarchical clustering algorithm gives tree like diagram. Decide 'k' (No. Of clusters) = 2 after analysing the tree diagram (Distribution of users, How far are the clusters?).
6. Calculate average scores for each cluster, Analyse average scores of clusters to find which dimensions differ most in average and deviation of participants of cluster from the average for those dimensions. Identify dimensions which are important for defining persona/cluster from above analysis. Identify two typical users, one from each cluster which shows minimum deviation from the average value.
7. Create 2 detailed user profiles from above two selected users and refine those profiles as more information about them is available.

3. Collaborative Filtering Auto-encoder implementation: https://towardsdatascience.com/deep-autoencoders-for-collaborative-filtering-6cf8d25bbf1d

Collaborative Filtering is a method used by recommender systems to make predictions about an interest of a specific user by collecting taste or preferences information from many other users. The technique of Collaborative Filtering has the underlying assumption that if a user A has the same taste or opinion on an issue as the person B, A is more likely to have B's opinion on a different issue.

An Autoencoder is an artificial neural network used to learn a representation (encoding) for a set of input data, usually to achieve dimensionality reduction. Architecturally, the form of an Autoencoder is a feedforward neural network having an input layer, one hidden layer and an output layer. The output layer has the same number of neurons as the input layer for the purpose of reconstructing it's own inputs. It is useful that an Autoencoder has a smaller hidden layer than the input layer. This effect forces the model to create a compressed representation of the data in the hidden layer by learning correlations in the data.

The extension of the simple Autoencoder is the Deep Autoencoder. The additional hidden layers enable the Autoencoder to learn mathematically more complex underlying patterns in the data. The first layer of the Deep Autoencoder may learn first-order features in the raw input. The second layer may learn second-order features corresponding to patterns in the appearance of first-order features.

To put everything together: We need additional layers to be able to handle more complex data — such as the data we use in collaborative filtering.

4. Training Deep Autoencoders for Collaborative Filtering: https://arxiv.org/pdf/1708.01715.pdf

Recommender systems can be divided into two categories: context-based and personalised recommendations. Context based recommendations take into account contextual factors such as location, date and time [1]. Personalised recommendations typically suggest items to users using the collaborative Filtering (CF) approach.

This is a classic CF problem: Infer the missing entries in an mxn matrix, R, whose (i, j) entry describes the ratings given by the ith user to the jth item. e performance is then measured using Root Mean Squared Error (RMSE).

Iterative output re-feeding - a technique which allowed us to perform dense updates in collaborative Filtering, increase learning rate and further improve generalisation performance.

5. Collaborative Filtering Neural Network Implementation: https://towardsdatascience.com/paper-review-neural-collaborative-filtering-explanation-implementation-ea3e031b7f96

Matrix Factorisation:

Utility matrix: user-item matrix used in CF.
It decomposes the utility matrix into two sub matrices. During prediction, we multiply the two sub-matrices to reconstruct the predicted utility matrix. The utility matrix is factorised such that the loss between the multiplication of these two and the true utility matrix is minimised. One commonly used loss function is mean-squared error.

Essentially, each user and item is projected onto a latent space, represented by a latent vector. The more similar the two latent vectors are, the more related the corresponding users' preference. Since we factories the utility matrix into the same latent space, we can measure the similarity of any two latent vectors with cosine-similarity or dot product. In fact, the prediction for each user/ item entry is computed by the dot product of the corresponding latent vectors.

Neural CF:

In the input layer, the user and item are one-hot encoded. Then, they are mapped to the hidden space with embedding layers accordingly. Here, embedding layers are latent vectors of user and item.

Embedding layers are given as input to either Neural Network layers like Multi-Layer Perceptron (MLP) OR to Generalised Matrix Factorisation layer (GMF) with fixed weight matrix with all 1's at output layer.

NeuMF:

Above embedding layers are given input to both GMF and MLP and concatenated results from both.

6. Associative Rule Mining: https://pdfs.semanticscholar.org/b298/e06b9ee4b3056c68a023035f228527a891a2.pdf

The system uses data mining techniques to discover a set of rules describing customers' behaviour and supports human experts in validating the rules.

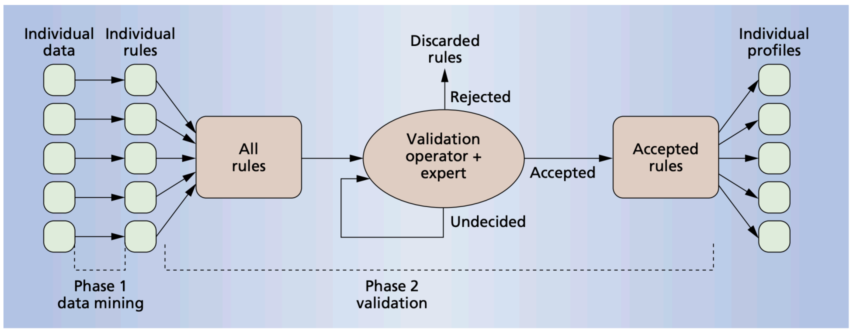
We have developed an approach that uses information learned from customers' transactional histories to construct accurate, comprehensive individual profiles.7 One part of the profile contains facts about a customer, and the other part contains rules describing that customer's behaviour. We use data mining methods to derive the behavioural rules from the data. We have also developed a method for validating customer profiles with the help of a human domain expert who uses validation operators to separate "good" rules from "bad."

Data Model:
- Factual (who the customer is): demographic information such as name, gender, birth date, address, salary, and social security number
- Transactional (what the customer is): records of the customer's purchases during a specific period. A purchase record might include the purchase date, product purchased, amount paid, coupon use, coupon value, and discount applied

Profile Model:
- Factual: the personalisation system obtained from the customer's factual data. The factual profile also can contain information derived from the transactional data, such as "The customer's favourite beer is Heineken" or "The customer's biggest purchase last month was for $237."
- Behavioural: models the customer's actions and is usually derived from transactional data. Examples of behaviours are "When purchasing cereal, John Doe usually buys milk" and "On weekends, John Doe usually spends more than $100 on groceries."

Rule Discovery:
We model individual customer behaviour with various types of conjunctive rules, including association and classification rules. We apply rule discovery methods individually to every customer's data.
- Associative rules: Apriori algorithm
- Classification rules: CART (Classification and Regression Trees)

Rule Validation:
Generally, Domain experts does the validation but following rule operators can be used to accept or reject rules.
- Similarity based rule grouping
- Template based rule filtering
- Redundant rule elimination



7. Word embeddings for food and recipes: https://jaan.io/food2vec-augmented-cooking-machine-intelligence/

Food Similarity:
- After training the embedding algorithm on a collection of 95,896 recipes, we get 100 dimensional embeddings for each food.
- We can calculate food similarity by looking at which food is closest in the high dimensional space in the embeddings.

Recipe Embedding:
- We can generate an embedding for a recipe by taking the average of its ingredients' embeddings.
- Interesting patterns emerge. Asian recipes cluster together, as do Southern European recipes. Northern European and American foods are all over the place, maybe because of transmission of recipes due to migration, or over-representation in the data.

Recipe Recommendation:
- By taking the average embedding for a set of foods, we can look up with the closest embeddings.
- Tool Demo: **https://altosaar.github.io/food2vec/#recipe-recommendation-tool**
- Github Link of implementation: **https://github.com/altosaar/food2vec**