Question 1. What kind of algorithms would you explore to solve this issue? What's your preferred model? How would you compare different models, and why? Explain the pros and cons of each of these models.

Answer: Best algorithm and model to create a real-time recommendation system in my opinion will be **Model based Low-Rank Matrix factorization for Collaborative Filtering**. Other models for this are:

1. Content-based recommenders, the general idea behind these recommender systems is that if a person likes a particular item, he or she will also like an item that is like it. And to recommend that, it will make use of the user's past item metadata.

Pros: The model doesn't need any data about other users since the recommendations are specific to this user. The model can capture the specific interests of a user.

Cons: New users or items with previously unseen features will logically suffer from this. The model can only make recommendations based on existing interests of the users so has limited ability to expand on the users' existing interests.

2. Memory based approach: Directly works with values of recorded interactions, assuming no model, and are essentially based on nearest neighbour's search (for example, find the closest users from a user of interest and suggest the most popular items among these neighbours).

Pros: The quality of predictions are rather good. This is a relatively simple algorithm to implement for any situation.

Cons: As no training or optimization is involved, it is an easy-to-use approach. But its performance decreases when we have sparse data which hinders scalability of this approach for most of the real-world problems. It uses the entire database every time it makes a prediction, so it needs to be in memory it is very, very slow.

Question 2. Let's assume we have chosen to work with a matrix factorization model.

- What are the steps/techniques you use to make sure that you're not over-fitting your model?

  Answer: The problem can be solved using ALS (Alternating Least Squares) or stochastic gradient descent to **perform regularization which prevents Overfitting and Bias ()** [2].

  Formulation of matrix factorization to perform regularization:

$$\underset{\mathbf{U}^{(q)},\mathbf{V}^{(q)}}{\text{minimize}} ||P_{\Omega}(\mathbf{R}^{(q)} - (\mathbf{U}^{(q)})^{\top}\mathbf{V}^{(q)})||_F^2 + \lambda(||\mathbf{U}^{(q)}||_F^2 + ||\mathbf{V}^{(q)}||_F^2)$$

Equation 1: Low-Rank Matrix Factorization (SVD algorithm)

$P_{\Omega}(X)$ is the matrix with only the indices in Ω (set of observed ratings). $\| \cdot \|$ denotes the Forbenius 2-norm and λ is the regularization parameter for avoiding overfitting while estimating the user-item matrix. Uses 10-fold cross validation to select the value of λ. It takes one-fold and compare with 9 others and does it iteratively for all the folds. SVD algorithm handles bias by make use of features of the data to minimize the error between the actual value and the predicted value.

- What techniques would you use to detect outliers?

  Answer: Outliers in online activity may exist from a few visitors that randomly clicked or bought too many items on a particular day. We can detect outliers from other users using total click and buy activities per day by a customer, within a time range that captures a significant amount of data. The process eliminates outliers on a particular day from the calculation of two metrics: **Click through rate (CTR)**, **Buy-through rate (BTR)** and **Add-to cart-through rate (ATC-TR)**.

- How would you solve the cold start problem? (i.e., how would you update the algorithm so it not only can make recommendations to the existing users in the recommender, but also to new users)

  Answer: Cold start problems can be handled by recommendations based on meta information, such as:

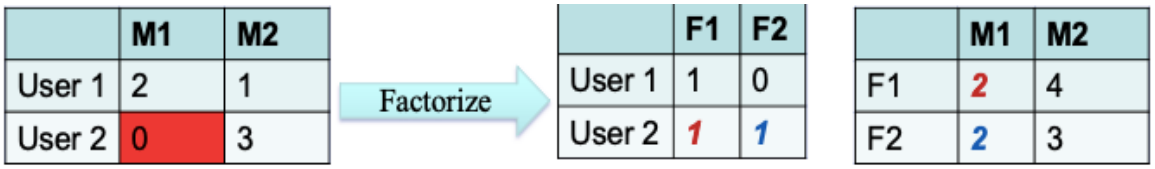
  a). For new users, we can use their Location, Age, Gender, Browser, and user device to predict recommendations. Also displaying popular items from the start.

  b). For existing users, we can use their historical data of previously purchased products and utilize that information to recommend similar products with similar product attributes such as: Color, Size, Type, Price. Also, identifying what other similar users with similar interests purchase to recommend to existing users.

- Propose a model that incorporates both the ratings and the Shoppers and Products attributes (age, gender, location for shoppers, type, size, color for products)? Describe your technique.

  Answer: The Singular Value Decomposition (SVD), is a matrix factorization technique. The SVD is used as a collaborative filtering technique. It uses a matrix structure where each row represents a **user (Shoppers)**, and each column represents an **item (Products)**. The elements of this matrix are the **ratings** that are given to items by users. The factorisation of this matrix is done by the singular value decomposition. SVD decomposes three matrices and the latent factors show the **products attributes**. Steps in the SVD to incorporate (ratings, Shoppers and product's attributes)

  Step 1: First SVD performs matrix factorization to produce two smaller matrices, one representing users (Shoppers) and one representing items (Products), which when multiplied together will produce roughly this matrix of ratings. Below image is taken from my thesis.



Rating matrix R | User Matrix | Item matrix

Suppose we take rating matrix R with two product attributes F1 and F2 in user matrix and item matrix where F1="age" and F2=" location". Also, M1 and M2 being products.

Question 3. Assume we have 1 instance of our model per store (each store has its own recommender) due to resource (memory and time) limitations. How would you efficiently recommend items from one store to another? These stores can share shoppers. Describe your solution(s).

Answer:

1). We can integrate stores via web services to identify commonalities between multiple stores such as common shoppers and can keep track of user behaviour, analyze product attribute they interact with to integrate stores. By integrating stores, shoppers are automatically stored across

multiple devices as well and we can use shoppers' information to recommend items from another store as well as from the same store.

References:

[1] A. Argyriou, M. González-Fierro, and L. Zhang, "Microsoft Recommenders: Best Practices for Production-Ready Recommendation Systems", WWW 2020: International World Wide Web Conference Taipei, 2020

[2] Liu, Xinyue & Kong, Xiangnan & Yu, Philip. (2018). Active Opinion Maximization in Social Networks. 1840-1849. 10.1145/3219819.3220061.