Assignment 2

Problem Statement:

Predict the list of two other items/products which are frequently bought together whenever an item/product is added to the cart.

Overview:

We will convert the given data into vectors, and the relative distance between them will essentially represent the similarity between the items. For example, we expect to find the vectors representing bread and butter to be closer, compared to say bread and wall decoration. And thus, we can predict the items frequently bought together by vectorizing the data.

Method:

1. Data Cleaning:
   We first clean the data so that we can filter the invalid/unavailable data, and obtain the important information to feed into the model.
   - Duplicate Entries: There are more than 50% entries which are duplicated. We remove these entries.
   - UserId: We have ~25% entries with invalid UserId (-1). However, we decide to keep these entries as we will be utilizing TransactionId as a means of identifying unique transactions, and we do not need to map these transactions to the users.
   - ItemCode: Invalid ItemCode (-1) entries are removed as the description provided indicated transactions related to non-purchasable items.
   - NumItemsPurchased: Assuming that negative numbers in this column represent items not purchased or items returned, we remove these entries.
   - ItemDescription: We remove entries with no item description.

2. Data Preparation:
   We arrange the data in a list of list format, which can be passed as an input to Word2Vec. We create a list which has details of all the items bought in each transaction. For example, if we had a total of 3 transactions and the items bought would have been T1 - A, B, C; T2 - B, D, E, F; T3 - A, C, D, F; then the list prepared would be as follows: [[A, B, C], [B, D, E, F], [A, C, D, F]].
   In our case, this list has around 20,000 sublists, one for each transaction.

3. Model Summary:
   We choose a model in which each vector will have a size of 50.
   The **skip-gram** model is chosen as it is used to predict the context from words.
   After training the model, we get a model which has learned 3043 words, each being represented by a vector of size 50.

4. Visualization:
   As we cannot visualize how the vectors appear in 50-dimensional space, we ‘flatten’ these vectors from 50 dimensions to 2D space using umap. Then we can plot the data and roughly see the clusters of items.

Note:

We are training the model on **ItemCodes**, i.e. we converting the list of ItemCodes to vectors instead of **ItemDescriptions**. This might seem counter-intuitive as we would generally want to train the model on text data. But we can get away with this as we are just trying to find out the co-occurrence of items, and since ItemCode and ItemDescription have a one-to-one correspondence, most likely the description is not going to give us any additional significant insights. Also, by making this change, we reduce the time taken to train the model as the training data size is concise.