1. MAP : Mean Average Precision
    1. MAP@K gives insight into how relevant the list of recommended items are

$$\text{recommender system precision:} \qquad P = \frac{\#\text{ of our recommendations that are relevant}}{\#\text{ of items we recommended}}$$

    2. How to decide which are relevant recommendations???
    3. Here, Recommendations that are relevant can be recommended recipes that user clicked
    4. We can calculate MAP to evaluate auto-encoder results too, where relevant recommendations can be products that were actually added by user out of all the products that auto-encoder predicted.
2. MAR : Mean Average Recall
    1. MAR@K gives insight into how well the recommender is able to recall all the items the user has rated positively in the test set.

$$\text{recommender system recall:} \qquad r = \frac{\#\text{ of our recommendations that are relevant}}{\#\text{ of all the possible relevant items}}$$

    2. Here, numerator can be calculated similarly as MAP
    3. No. Of all possible relevant items ???
3. Coverage: the percent of items in the training data the model is able to recommend on a test set
    1. If 250 out of 620 products have appeared in purchase history of users in training data, we can calculate how many out of these 250 were appeared in recommendations in testing data. This can be calculated on Auto-Encoder and Associative Rule Mining output.
    2. We cannot apply this on recipe recommendation output as in our case we are predicting something different than what we are using for the predictions. Eg. We have used purchase history of ingredients and recommending recipes. But in MovieLens dataset, movies are recommended from ratings given to the movies.
4. Personalisation:
    1. Dissimilarity (1- cosine similarity) between recipes recommended to users in the test data.
    2. Here, we can encode list of recommendation into a vector where vector length will be total no. of recipes in the database and value in the vector will be one if corresponding recipe is recommended. Then we can calculate cosine similarity of above vectors.
    3. Can we use Jaccard Similarity also?
    4. We can analyse if users assigned to different clusters show more dissimilarity that users assigned to the same cluster in the test data.
5. Intra-List Similarity:
    1. The average cosine similarity of all items in a list of recommendations. This calculation uses features of the recommended items (such as movie genre) to calculate the similarity
    2. In our case, For each recipe in recommendation list, perform One Hot Encoding on it where columns of the vector are all cuisines in the dataset and value is 1 if recommended recipe belongs to the cuisine.
    3. Calculate cosine similarity of each recipe with other recipes in the recommendation list.
    4. Take average of all cosine similarities calculated above which will represent Intra-List Similarity for that particular user's recommendation.
    5. Intra-list similarity for the model: We can calculate average Intra-List Similarity for all users in test set by taking average of Intra-List Similarity score of all users in the test set calculated in the above step.
    6. If a recommender system is recommending lists of very similar recipes to single users (for example, a user receives only recommendations of Italian cuisine), then the intra-list similarity will be high.