**Recommender Systems.**

Recommender systems are among the most popular applications of data science today. They are used to predict the "rating" or "preference" that a user would give to an item.Almost every major tech company has applied them in some form. Amazon uses it to suggest products to customers, YouTube uses it to decide which video to play next on autoplay, and Facebook uses it to recommend pages to like and people to follow.

There are also popular recommender systems for domains like restaurants, movies, and online dating. Recommender systems have also been developed to explore research articles and experts, collaborators, and financial services.

**Broadly, recommender systems can be classified into 3 types:**

**Simple recommenders:** offer generalized recommendations to every user, based on movie popularity and/or ratings.

**Popularity problems:** Aging (something that was popular a long time ago might not be desired at the moment), generalization (one might prefer something different depending on their situation such as a tourist might prefer local restaurant to a popular Macdonalds’)

**Solution**: Use age to penalize the older items such that score decreases with age and increasing with popularity. So that newer items can be on top.

**Average Ratings problems 1:** A rating can be high because there are few ratings, compared to an item with lower rating where it is rated by customers. Fewere ratings are also not reliable. Confidence with the ratings is important. Confidence interval for average ratings with few number of ratings(sample size) are wider, whereas confidence intervals for average ratings with large number of ratings are narrow indicating that we are more confident with large number of ratings.

**Solution 1.** Rate each item by the lower bound of the confidence interval.

**Average Ratings problems 2:** When we have zero ratings for the item, the average is undefined.

**Solution 2.** Smoothing or Dumpening, result in a default value for items with no rating such as a global mean,median or any chosen number, by adding a small number to the numerator and the denominator of the average(mean) formula.

**Content-based recommenders:** suggest similar items based on a particular item. This system uses item metadata, such as genre, director, description, actors, etc. for movies, to make these recommendations. 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 similar to it. And to recommend that, it will make use of the user's past item metadata. A good example could be YouTube, where based on your history, it suggests you new videos that you could potentially watch.

**Collaborative filtering engines:** these systems are widely used, and they try to predict the rating or preference that a user would give an item-based on past ratings and preferences of other users. Collaborative filters do not require item metadata like its content-based counterparts.

## **Collaborative Filtering**

Collaborative filters can further be classified into two types:

**User-based Filtering:** these systems recommend products to a user that similar users have liked.For example, let's say Alice and Bob have a similar interest in books (that is, they largely like and dislike the same books). Now, let's say a new book has been launched into the market, and Alice has read and loved it. It is, therefore, highly likely that Bob will like it too, and therefore, the system recommends this book to Bob.

**Item-based Filtering:** these systems are extremely similar to the content recommendation engine that you built. These systems identify similar items based on how people have rated it in the past. For example, if Alice, Bob, and Eve have given 5 stars to *The Lord of the Rings* and *The Hobbit*, the system identifies the items as similar. Therefore, if someone buys *The Lord of the Rings*, the system also recommends *The Hobbit* to him or her. Item-based filtering performs better that user-based filtering because items have more data than users and result in more accurate weights.

**Limitations of CF:**

Resulting in poor recommendation accuracy and reduced coverage

1. **Cold-start problems,** if we do not have enough data we cannot calculate correlations

**Solution:** If there is no data at all, add a prior to the average, score can be a weighted sum of prediction plus prior average. To get a prior, scrape from the web(one possibility). Use a Bayesian approach

1. **Data Sparsity,**explicit ratings are sparse, many users do not rate items unless they really liked them or hated them.

**Solution:**Use implicit measure such as

# **Developing recommender systems with the consideration of product profitability for sellers.**

This study attempts to integrate the profitability factor into the traditional recommender systems. Based on this consideration, we propose two profitability-based recommender systems called ***CPPRS* (*Convenience plus Profitability Perspective Recommender System*)** and ***HPRS* (*Hybrid Perspective Recommender System*)**. Moreover, comparisons between our proposed systems (considering both purchase probability and profitability) and traditional systems (emphasizing an individual’s preference) are made to clarify the advantages and disadvantages of these systems in terms of recommendation accuracy and/or profit from cross-selling. The experimental results show that the proposed HPRS can increase profit from cross-selling without losing recommendation accuracy.

# **A trust-semantic fusion-based recommendation approach for e-business applications.**

Collaborative Filtering (CF) is the most popular recommendation technique but still suffers from data sparsity, user and item cold-start problems, resulting in poor recommendation accuracy and reduced coverage. This study incorporates additional information from the users' social trust network and the items' semantic domain knowledge to alleviate these problems. It proposes an innovative Trust–Semantic Fusion (TSF)-based recommendation approach within the CF framework. Experiments demonstrate that the TSF approach significantly outperforms existing recommendation algorithms in terms of recommendation accuracy and coverage when dealing with the above problems. A business-to-business recommender system case study validates the applicability of the TSF approach.

**Reference**

1. **Datacamp**
2. **Udemy,**[**https://explore.udemy.com/course/recommender-systems/learn/lecture/11717456#questions**](https://explore.udemy.com/course/recommender-systems/learn/lecture/11717456#questions)