Project Proposal – Capstone Project # 2

# Project Idea 1: Build a Movie Recommendation Engine

## Problem statement

Recommend new movies to users.

## Objective

Many online businesses rely on customer reviews and ratings. Explicit feedback is especially important in the entertainment and ecommerce industry where all customer engagements are impacted by these ratings. Netflix relies on such rating data to power its recommendation engine to provide the best movie and TV series recommendations that are personalized and most relevant to the user.

## Data Description

I will be using the MovieLens dataset for this purpose. It has been collected by the GroupLens Research Project at the University of Minnesota. t consists of:

* 100,000 ratings (1-5) from 943 users on 1682 movies.
* Each user has rated at least 20 movies.
* Simple demographic info for the users (age, gender, occupation, zip)
* Genre information of movies

1. Rating file: All ratings are contained in the file "ratings.dat" and are in the following format:

UserID::MovieID::Rating::Timestamp

* UserIDs range between 1 and 6040
* MovieIDs range between 1 and 3952
* Ratings are made on a 5-star scale (whole-star ratings only)
* Timestamp is represented in seconds since the epoch as returned by time
* Each user has at least 20 ratings

1. Users file: User information is in the file "users.dat" and is in the following format:

UserID::Gender::Age::Occupation::Zip-code

All demographic information is provided voluntarily by the users and is not checked for accuracy. Only users who have provided some demographic information are included in this data set.

* Gender is denoted by a "M" for male and "F" for female
* Age is chosen from the following ranges:

\* 1: "Under 18"

\* 18: "18-24"

\* 25: "25-34"

\* 35: "35-44"

\* 45: "45-49"

\* 50: "50-55"

\* 56: "56+"

* Occupation is chosen from the following choices:

\* 0: "other" or not specified

\* 1: "academic/educator"

\* 2: "artist"

\* 3: "clerical/admin"

\* 4: "college/grad student"

\* 5: "customer service"

\* 6: "doctor/health care"

\* 7: "executive/managerial"

\* 8: "farmer"

\* 9: "homemaker"

\* 10: "K-12 student"

\* 11: "lawyer"

\* 12: "programmer"

\* 13: "retired"

\* 14: "sales/marketing"

\* 15: "scientist"

\* 16: "self-employed"

\* 17: "technician/engineer"

\* 18: "tradesman/craftsman"

\* 19: "unemployed"

\* 20: "writer"

1. Movies file: Movie information is in the file "movies.dat" and is in the following format:

MovieID::Title::Genres

* Titles are identical to titles provided by the IMDB (including year of release)
* Genres are pipe-separated and are selected from the following genres:

\* Action

\* Adventure

\* Animation

\* Children's

\* Comedy

\* Crime

\* Documentary

\* Drama

\* Fantasy

\* Film-Noir

\* Horror

\* Musical

\* Mystery

\* Romance

\* Sci-Fi

\* Thriller

\* War

\* Western

* Some MovieIDs do not correspond to a movie due to accidental duplicate entries and/or test entries
* Movies are mostly entered by hand, so errors and inconsistencies may exist

## Approach

# This problem will be solved using Recommendation Algorithms. There are typically two types of algorithms – Content Based and Collaborative Filtering:

1. **Content based algorithms:**

* Idea: If you like an item then you will also like a “similar” item
* Based on similarity of the items being recommended
* It generally works well when it’s easy to determine the context/properties of each item. For an instance when we are recommending the same kind of item like a movie recommendation or song recommendation.

1. **Collaborative filtering algorithms:**

* Idea: If a person A likes item 1, 2, 3 and B like 2,3,4 then they have similar interests and A should like item 4 and B should like item 1.
* This algorithm is entirely based on the past behavior and not on the context. This makes it one of the most commonly used algorithms as it is not dependent on any additional information.
* For instance: product recommendations by e-commerce player like Amazon and merchant recommendations by banks like American Express.
* Further, there are several types of collaborative filtering algorithms:

1. **User-User Collaborative filtering:** Here we find look alike customers (based on similarity) and offer products which first customer’s look alike has chosen in past. This algorithm is very effective but takes a lot of time and resources. It requires to compute every customer pair information which takes time. Therefore, for big base platforms, this algorithm is hard to implement without a very strong parallelizable system.
2. Item-Item Collaborative filtering: It is quite similar to previous algorithm, but instead of finding customer look alike, we try finding item look alike. Once we have item look alike matrix, we can easily recommend alike items to customer who have purchased any item from the store. This algorithm is far less resource consuming than user-user collaborative filtering. Hence, for a new customer the algorithm takes far lesser time than user-user collaborate as we don’t need all similarity scores between customers. And with fixed number of products, product-product look alike matrix is fixed over time.
3. Other simpler algorithms: There are other approaches like market basket analysis, which generally do not have high predictive power than the algorithms described above.

## References

<https://www.analyticsvidhya.com/blog/2016/06/quick-guide-build-recommendation-engine-python/>

# Project Idea 2: Instacart Market Basket Analysis

## Problem statement

Which products will an Instacart consumer purchase again?

## Objective

The client is Instacart

The overall objective is to predict products that a user will buy again, try for the first time or add to cart next during a session.

Instacart currently uses XGBoost, word2vec and Annoy in production on similar data to sort items for users to “buy again”.

This data, and the algorithms trained upon it, are enabling Instacart to revolutionize how consumers discover and purchase groceries.

This helps Instacart make the right product recommendations to the customer, thereby, making the shopping experience more convenient for consumer.

## Data Description: “The Instacart Online Grocery Shopping Dataset 2017”

The dataset for this competition is a relational set of files describing customers' orders over time. The goal of the competition is to predict which products will be in a user's next order. The dataset is anonymized and contains a sample of over 3 million grocery orders from more than 200,000 Instacart users. For each user, we provide between 4 and 100 of their orders, with the sequence of products purchased in each order. We also provide the week and hour of day the order was placed, and a relative measure of time between orders.

File descriptions

1. orders (3.4m rows, 206k users):

* order\_id: order identifier
* user\_id: customer identifier
* eval\_set: which evaluation set this order belongs in (see SET described below)
* order\_number: the order sequence number for this user (1 = first, n = nth)
* order\_dow: the day of the week the order was placed on
* order\_hour\_of\_day: the hour of the day the order was placed on
* days\_since\_prior: days since the last order, capped at 30 (with NAs for order\_number = 1)

1. products (50k rows):

* product\_id: product identifier
* product\_name: name of the product
* aisle\_id: foreign key
* department\_id: foreign key

1. aisles (134 rows):

* aisle\_id: aisle identifier
* aisle: the name of the aisle

1. departments (21 rows):

* department\_id: department identifier
* department: the name of the department

1. order\_products\_\_SET (30m+ rows):

* order\_id: foreign key
* product\_id: foreign key
* add\_to\_cart\_order: order in which each product was added to cart
* reordered: 1 if this product has been ordered by this user in the past, 0 otherwise

where SET is one of the four following evaluation sets (eval\_set in orders):

* "prior": orders prior to that users most recent order (~3.2m orders)
* "train": training data supplied to participants (~131k orders)
* "test": test data reserved for machine learning competitions (~75k orders)

<https://tech.instacart.com/3-million-instacart-orders-open-sourced-d40d29ead6f2>

## Approach

* This is a supervised learning algorithm because we have the training data
* And this is a classification problem because we have fixed number of categories we need to predict as part of the response variable

The variable to predict is product that an instacart consumer will purchase

The predictor variables can be as follows: Order day of week , order hour of day, days since prior order, aisle, department, 'reordered' which indicates that the customer has a previous order that contains the product.

## Citation

“The Instacart Online Grocery Shopping Dataset 2017”, Accessed from https://www.instacart.com/datasets/grocery-shopping-2017 on 3/8/2020

## Deliverables

The following list of deliverables need to be created as part of the project output:

* Code
* Presentation containing an executive summary and the approach followed