Recommender System Architectural Outline – NetFlix Data Set

***Green highlights denote direction of recommendation system for simplicity and time sake = down select.

- Recommender systems have been around since the 90's like amazon's book recommendation system
- Goals of recommendation systems (two-fold):
    1. **Predictions – for simplicity and time sake we are going to look at predictions**
    2. Ranking version – this can be complex for this short period we are going to concentrate on predictions
- Operational and technical goals of a recommender system
    1. Relevance -relevant to the user at hand; users are more likely to consume something they find interesting
    2. Novelty -helpful in seeing or using something it has not done so in the past
    3. Serendipity – lucky discovery
    4. Increasing recommendation diversity – suggesting a list of top items that are very similar
- Netflix dataset provides recommendations based on a five star rating – 5 point scale
- Basic Models of Recommender Systems:
    1. Collaborative Filtering – power of the ratings provided by multiple users to make recommendations. Challenge: use of sparse matrices
        - Memory based methods:
            1. User based collaborative filtering – ratings provided by "like-minded" individuals of a target user A are used to make recommendations for A, and recommend ratings for the unobserved ratings of A by computing weighted averages of the ratings of this peer group – too complex!
            2. Item based collaborative filtering – Make the rating predictions for target item B by user A, the first step is to determine a set S of items that are most similar to target item B. The ratings in item set S, which are specified by A are used to predict whether the user A will like item B.
        - Advantages of memory-based techniques are that they are simple to implement and the resulting recommendations are often easy to explain.
        - Model based methods are used in the context of machine learning and data mining for predictive models.
- Ratings – 5 = Loved it!; 4 = I liked it!; 3= it was ok; 2= I didn't like it; 1= I hated it! 0 = no opinion or didn't rate it
- Collaborative filtering as a generalization of classification and regression modeling

- Content based recommender systems – descriptive attributes of items used to make recommendations – more NLP related
- Knowledge based recommender systems- useful in the context of items that are not purchased very often
- Demographic recommender systems
- Hybrid and ensemble based recommender systems

Neighborhood based collaborative filtering

- **important distinction between user based collaborative filtering and item-based collaborative filtering algorithms is that ratings in the former case are predicted using the rating of the neighboring users, whereas the ratings in the latter case are predicted using the user's own ratings on neighboring items.

- We are going with user-based collaborative filtering for easy and expedience of implementation.
- User-based models: Similar users have similar ratings on the same item. Therefore if Bob and Joe have rated movies in a similar way in the past, then one can use Alice's observed ratings on the movie Alien to predict Bob's unobserved ratings on this movie.
- Item-based models: Similar items are rated in a similar way by the same user. Theefore bob's ratings on similar science fiction movies like Alien and Predator can be used to predict his rating on Terminator.
    1. Item based models using adjusted cosine – results in the computation of mean centered matrix – normalized to a mean of 0
    2. Fine the top k matching movies most similar to items based on item X that you are recommending for Q.
    3. Leverage the user's own ratings on similar items in the final step of making the prediction. –remember it's an item "peer group"
    - Strengths of Neighborhood Based Methods
        1. Simplicity and intuitive
        2. Recommendations are stable with the addition of new items and users
    - Disadvantages of Neighborhood Based Methods
        1. If nearest neighbors of John did not rate the terminator makes it very hard to recommend the terminator for John; sparcity creates challenges for similarity computation when the number of mutually rated items between two users are small
    - Dimensionality reduction can be used to improve neighborhood based methods in terms of quality and efficiency such as PCA
    - Possible to use decision tree and regression trees or Naïve Bayes collaborative filtering or deep learning

Execution - simple

- Utilize pandas and numpy
- Load data through csv
- Group ratings by title and rating – calculate mean here
- Bucket the number of ratings per movie→ groupby title and rating.count()
- We could do a plot to visualize the distribution of ratings
- Create our movie matrix (pivot table) → index user id, columns are title and rating
- Sort ratings
- We could then use the Corwith function in pandas to calculate the correlaton between two data frames

Execution – more complex but still fairly easy collaborative filterting using kNN

Content based approach utilizes a series of discrete characteristics of an item in order to recommend additional items with similar properties.

Collaborative filtering approach builds a model from a user's past behaviors (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that the user may have an interest in.

Hybrid approach combines the previous two approaches. Most businesses probably use hybrid approach in their production recommender systems. – if time allows…..means we need to architect in such a way to code it in easily

Data: We need user id, movie id, rating, date

Packages: sklearn and pandas, scipy for sparse matrices

kNN is perfect for itembased approached!!!

- Create a pivot table for index – movie id
    - Columns user id and rating
    - Ensure we select cosine similarity for nearest neighbor search or locality sensitive hashing. Euclidean distance if selected will have to do PCA – cosine similarity is more exact.

Software Architecture

- Class knn recommender—
- Function Filter parameters – set rating frequency and threathold to folter out less known movies and less active users
- Function for model parameters – set them for sklearn

- Function for data prep→ movie user sparse matrix and a map of movie to row index using a sparse matrix
- Function for fuzzy mapping to map movie title name to index of movie in data to return index of closest matched movie
- Function for inference → return top XXX similar movie recommendations based on users input movie -- **movie must be in the database
- Parameterize it by movie name and top XXXX maybe for selection in GUI?
    - If select movie name then selects movies most similar rating to the movie name in the DB
    - Or select top XXXX similar titles to “Charlottes Web” – maybe top 10?
    - Movie input in gui
- Function for making recommendations utilizing a sparse matrix → makes top movie recommendations