1 Recommendation Steps

1. List set W_u of movies watched by user U_i, extract tag-genome matrix for W_u.
2. Calculate Silhouette scores for different possible cluster sizes, different sizes are from 2 to $|W_u|$.
3. Select $n\_clusters$ such that it yields the highest Silhouette score.
4. Rank clusters by their sizes, i.e. number of movies in each cluster. Rank in such a fashion that highest size cluster gets the low rank. These ranks are used by the 'Novel Ranking' algorithm.
5. **N_Similar_movies** is a configurable parameter that controls the number of movies in Recommendation List **RL** that are most similar to the user profile. Select *N_Similar_movies* from the dense clusters, using **Algorithm 1**. Dense clusters are the clusters >= mean cluster size.
6. In case, above step returns less number of movies, the difference should be covered by the next movie selection **Algorithm 2**. Hence, recalculate the value of **N_Novel_Movies** as:

$$N\_Novel\_Movies = K - N\_Similar\_Movies \tag{1}$$

7. Find all possible N_Novel_Movies using the **Algorithm 2**.
8. Append N_Similar_Movies to the Recommendation List **RL**.
9. Choose N_Novel_Movies from the Novel Recommendation List **NRL**, and append them to the final Recommendation List **RL**.
10. Recommend the Recommendation List **RL**.

Algorithm 1 Most Similar Movies Selection Algorithm

$D \leftarrow dense\_clusters$ ▷ D is a set of Dense Clusters, input argument
$N \leftarrow N\_Movies\_Per\_Dense\_Cluster$ ▷ Input argument N
while D *has more clusters* **do**
    $C \leftarrow next\_cluster\ from\ D$
    $W \leftarrow list\ of\ watched\ movies\ from\ C$
    $similar\_movies \leftarrow top\ similar\ movies$ ▷ Cosine similarity
    $RL1 \leftarrow$ append top N movies most similar to user's profile
return RL1

Algorithm 2 Novel Re-Ranking Algorithm

$W_u \leftarrow$ watched movies from all sparse clusters
$R_{df} \leftarrow$ A dataframe object used for Ranking
while W_u has more movies **do**
    $W_i \leftarrow$ next movie
    $R_{df} \leftarrow$ append similar N movies
    $R_{df}[C_i] \leftarrow$ cluster score for W_i
    $R_{df}[S_{wi}] \leftarrow$ similarity with W_i
$R_{df}[S_u] \leftarrow$ similarity of all movies to user profile
$R_{df}[R_{wi}] \leftarrow$ rating of user i for watched movie W_i
$R_{df}[diversity] \leftarrow$ 1 - $R_{df}[S_u]$
▷ Diversity = 1 - sim(user_profile)
$R_{df}[rank(R_{cu})] \leftarrow$ Rank for column values R_{cu}
▷ Dense rank as available in Pandas
$R_{df}[rank(diversity)] \leftarrow$ Rank for column values $diversity$
$R_{df}[rank(S_u)] \leftarrow$ Rank for column values S_u
$R_{df}[rank(S_c)] \leftarrow$ Rank for column values S_c
$R_{df}[RN_c] \leftarrow$ Composite Rank using equation 2
$NRL \leftarrow$Sort movies in descending order based on $R_{df}[RN_c]$ - composite rank.
return NRL ▷ Sorted Novel Recommendation List

Components of the algorithm 2,

1. R_{wi}, is Rating given by the user to the watched movie.
2. S_u, is the Similarity to the user's profile, calculated using cosine similarity with the user's tag-genome based term vector.
3. $diversity$, is the diversity to the user profile, calculated as '1 - S_u'
4. C_i, is the score for the cluster which has a watched movie W_i.

This is the ranking equation:

$$\begin{aligned} \mathrm{R}_{df}[RN_{comp}] = &(R_{wi}\_weight * R_{df}[rank(R_{wi})])+ \\ &(diversity\_weight * R_{df}[rank(diversity)])+ \\ &(S_u\_weight * R_{df}[rank(S_u)])+ \\ &(C_i\_weight * R_{df}[rank(C_i)]) \end{aligned} \tag{2}$$

The $R_{df}[RN_{comp}]$ is a composite rank for a movie.