**Trial Model 1:**

In this model, the suggestions are based solely on the tropes in movies selected by the user. First of all, all the tropes corresponding to the all the movies selected by the user are identified. This is called User Tropes. For each of these tropes, a weight is assigned based on the number of times it occurred within the user selected movies. This weight is called User Popularity. Then for each and every movie which has at least one trope in common with the User Tropes, total User Popularity is found by summing up User Popularity corresponding to all the tropes it has in common with User Tropes. The movies having the highest User Popularity are then suggested.

Packages Used:

Numpy and Pandas of Python

Results:

Do they make sense? (Ramesh to check)

Pros:

1. Simple to implement
2. Helps to sense the usefulness of the attribute ‘trope’ in finding similar movies.
3. Highly customized for the user, no impact of overall popularity of tropes or related tropes.

Cons:

1. Only the movies which have tropes in User Tropes list are considered. It would be helpful to consider all closely related tropes to User Tropes list.
2. Over all popularity of the tropes is not considered.
3. Popularities were just based on number of movies a trope appeared in. Other factors like number of highly rated movies it appeared in etc…were not considered.

**Trial Model 2**

In this model, in addition to calculating User Popularity as explained in **Trial Model 1,** Overall Popularity is also calculated. For each and every trope, Over Popularity is defined as the number of movies it occurred in total. Then for each and every movie which has at least one trope in common with the User Tropes, total Overall Popularity is found by summing up Overall Popularity corresponding to all the tropes it has in common with User Tropes. Also, total Combined Popularity is found by summing up the product of Overall Popularity and User Popularity corresponding to each of the tropes it has in common with User Tropes. The movies having the highest Combined Popularity are then suggested.

Packages Used:

Numpy and Pandas of Python

Results:

Do they make sense? (Ramesh to check)

Pros:

1. Simple to implement
2. Helps to sense the usefulness of the attribute ‘trope’ in finding similar movies.
3. Customized for the user, no impact of related tropes.

Cons:

1. Only the movies which have tropes in User Tropes list are considered. It would be helpful to consider all closely related tropes to User Tropes list.
2. Popularities were just based on number of movies a trope appeared in. Other factors like number of highly rated movies it appeared in etc…were not considered.

**Next Steps:**

1. Identify related tropes through Text Analysis and Manual Exercise.
2. Consider related tropes as well in the model for generating suggestion.
3. Create Popularity Metrics based on (If the user selects show popular movies):
   1. Number of Highly rated movies a trope is present in. (or a weighted popularity based on rating)
   2. Number of Highly awarded movies a trope is present in. (or a weighted popularity based on number of awards)
4. Consider other parameters such as length of the movie, release year (Can ask user to enter the ranges for these, if not filled in, all the movies would be considered) while filtering for the movies.
5. Incorporate feedback loop based on user feedback on suggested movies (deletion of irrelevant movies by the user etc…)