## ReelRx Model Breakdown

ReelRx is built on a collaborative, item based filtering model. It uses a nearest neighbors model with cosine similarity to predict items (movies with ratings) a user may be interested in based on decisions (ratings applied to movies) made by other users.

The foundation of ReelRx is the MovieLens 1M dataset. This dataset provides 1 million ratings from 6000 users on 4000 movies. The relevant parts of this dataset are:

movies.csv

| | Unnamed: 0 | movie_id | title | genres |
|---|---|---|---|---|
| **0** | 0 | 1 | Toy Story (1995) | Animation\|Children's\|Comedy |
| **1** | 1 | 2 | Jumanji (1995) | Adventure\|Children's\|Fantasy |
| **2** | 2 | 3 | Grumpier Old Men (1995) | Comedy\|Romance |
| **3** | 3 | 4 | Waiting to Exhale (1995) | Comedy\|Drama |
| **4** | 4 | 5 | Father of the Bride Part II (1995) | Comedy |

ratings.csv

| | Unnamed: 0 | user_id | movie_id | rating | timestamp | user_emb_id | movie_emb_id |
|---|---|---|---|---|---|---|---|
| **0** | 0 | 1 | 1193 | 5 | 978300760 | 0 | 1192 |
| **1** | 1 | 1 | 661 | 3 | 978302109 | 0 | 660 |
| **2** | 2 | 1 | 914 | 3 | 978301968 | 0 | 913 |
| **3** | 3 | 1 | 3408 | 4 | 978300275 | 0 | 3407 |
| **4** | 4 | 1 | 2355 | 5 | 978824291 | 0 | 2354 |

The backbone of ReelRx is a movie / user ratings matrix:

```
final_dataset = ratings.pivot(index='movie_id',columns='user_id',values='rating')
```

```
final_dataset.head()
```

| user_id | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | ... | 6031 | 6032 | 6033 | 6034 | 6035 | 6036 | 6037 | 6038 | 6039 | 6040 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **movie_id** | | | | | | | | | | | | | | | | | | | | | |
| **1** | 5.0 | 0.0 | 0.0 | 0.0 | 0.0 | 4.0 | 0.0 | 4.0 | 5.0 | 5.0 | ... | 0.0 | 4.0 | 0.0 | 0.0 | 4.0 | 0.0 | 0.0 | 0.0 | 0.0 | 3.0 |
| **2** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 5.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **3** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **4** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 3.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 2.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| **5** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

5 rows × 6040 columns

One goal of ReelRx was to provide two forms of movie recommendations: “Popular” and “obscure” recommendations, with popularity being determined by number of ratings a movie has (more ratings means more popular). Accordingly, there is a question of what threshold of user ratings a movie should have to qualify as popular vs obscure in the model.

An analysis was performed on the ratings:

```
no_user_voted = ratings.groupby('movie_id')['rating'].agg('count')
```

```
no_user_voted.head()
```

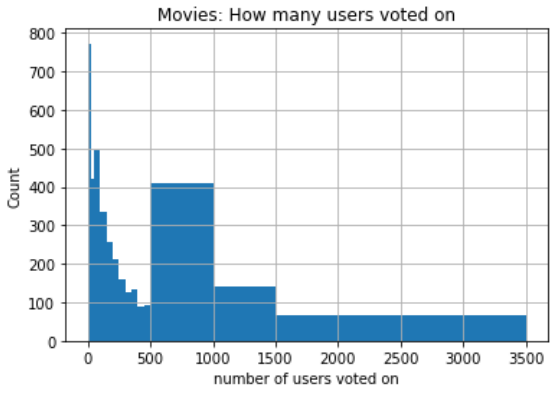
```
movie_id
1    2077
2     701
3     478
4     170
5     296
Name: rating, dtype: int64
```

```
no_user_voted.describe()
```

```
count    3706.000000
mean      269.889099
std       384.047838
min         1.000000
25%        33.000000
50%       123.500000
75%       350.000000
max      3428.000000
Name: rating, dtype: float64
```

```
no_user_voted.hist(bins=[0,25,50,100,150,200,250,300,350,400,450,500,1000,1500,3500])
plt.title("Movies: How many users voted on")
plt.ylabel("Count")
plt.xlabel("number of users voted on")
```

```
Text(0.5, 0, 'number of users voted on')
```

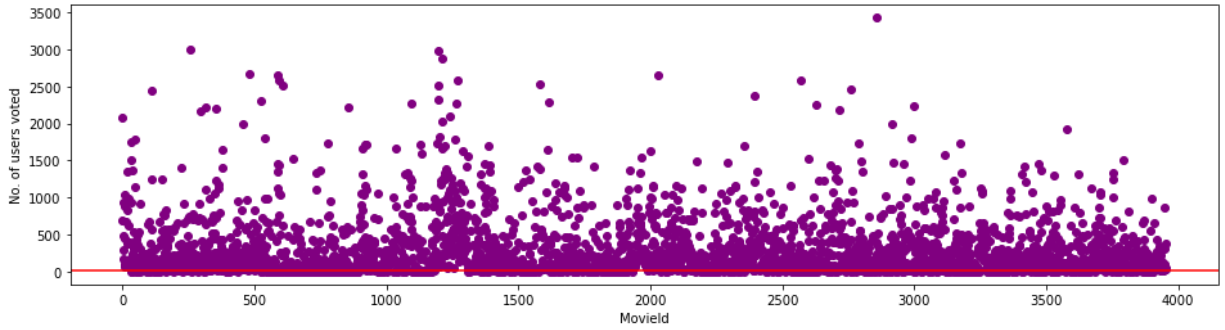


```
ptile = 20
p = np.percentile(no_user_voted,ptile)
print(p)
print(len(no_user_voted)*(ptile/100))
```

```
23.0
741.2
```

```
f,ax = plt.subplots(1,1,figsize=(16,4))

plt.scatter(no_user_voted.index,no_user_voted,color="purple")
plt.axhline(y=23,color='r')
plt.xlabel('MovieId')
plt.ylabel('No. of users voted')
plt.show()
```




The analysis resulted in the threshold between popular and obscure being the 20th percentile. This percentile includes movies with between 1 and 23 ratings and creates a population of 741 movies.

Movies with between 1 (minimum) and 23 (20th percentile) user ratings are considered “obscure” and those with greater than 23 user ratings are considered “not obscure”, or “popular”.

Each resulting dataset can be separately run through the model.

ReelRx applies a cosine similarity nearest neighbors model.

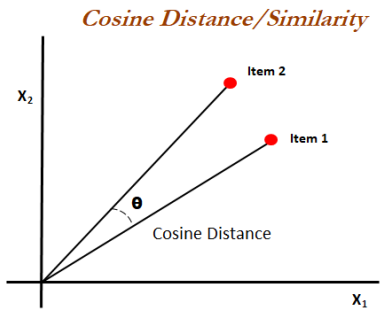
```
knn = NearestNeighbors(metric='cosine', algorithm='brute', n_neighbors=20, n_jobs=-1)
```

The chosen dataset is transformed into a Compressed Sparse Row matrix.

```
csr_data = csr_matrix(function_dataset.values)
```

```
knn.fit(csr_data)
```

The model uses cosine similarity. Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space.

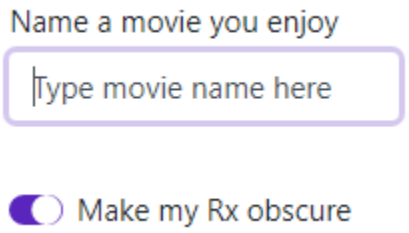



This metric was preferred over a euclidean distance model for two key reasons:

1. It runs with high speed and efficiency
2. Having the dataset in a matrix means that determining euclidean distances between groupings of user ratings produces results where ratings are often equidistant. Accordingly, a nearest neighbors matrix-based model fails to successfully predict in euclidean measurements, whereas it succeeds with angular measurements.

The ReelRx model's input is simple and two-fold:

An item (movie name)
A dataset (obscure or popular)




The output is a dataframe of the nearest neighbors to the entered item (movie) from either the obscure data set or the popular dataset, sorted on the distance (cosine similarity).

| Rx Score | *Click to learn more* |
|---|---|
| 0.5077554472 | Apollo 13 (1995) |
| 0.5062098547 | Ferris Bueller's Day Off (1986) |
| 0.5043727132 | Dave (1993) |
| 0.5021588993 | Wedding Singer, The (1998) |
| 0.5018734205 | When Harry Met Sally... (1989) |

The model successfully predicts recommendations for an inputting user based on other user behavior.

Many thanks to kaggle user johnwill225 for his legwork on this model.