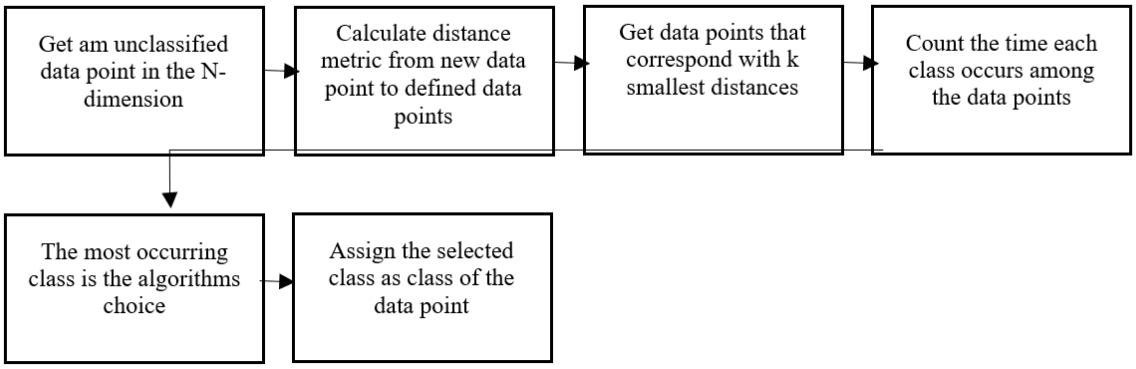
Planning Docs

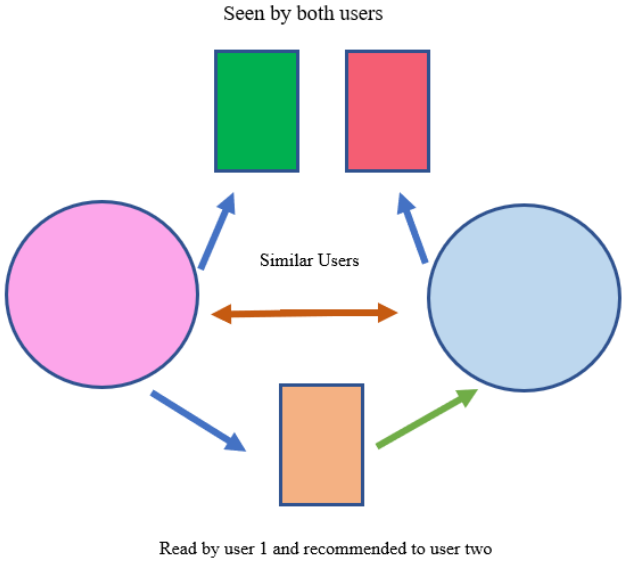
Framework Diagram:

Fig 2. Showing the KNN model



Software Methodology:

Fig 1. Showing the methodology and frame work for this prototype system.



Software Output:

```
In [4]: #Show headings for the rating dataset

        df_ratings.head()
```

Out[4]:

	userId	movieId	rating
0	1	307	3.5
1	1	481	3.5
2	1	1091	1.5
3	1	1257	4.5
4	1	1449	4.5

```
In [5]: # shaping the data rating setting at 200000
        df_ratings=df_ratings[:2000000]

In [6]: df_ratings.shape

Out[6]: (2000000, 3)
```

```
In [8]: mat_movie_features = csr_matrix(df_movie_features.values)

In [9]: df_movie_features.head()
```

Out[9]:

userId	1	2	3	4	5	6	7	8	9	10	...	20498	20499	20500	20501	20502	20503	20504	20505	20506
movieId																				
1	0.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	0.0	5.0	...	0.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	4.0	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	0.0	0.0
3	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
4	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	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	2.0	0.0	0.0	0.0	3.0	0.0	0.0	...	0.0	4.5	0.0	4.0	0.0	0.0	0.0	0.0	0.0

5 rows × 20507 columns

In [12]:
```
# getting the count
df_ratings_cnt_tmp = pd.DataFrame(df_ratings.groupby('rating').size(), columns=['count'])
df_ratings_cnt_tmp
```

Out[12]:

rating	count
0.5	37006
1.0	63892
1.5	31587
2.0	134360
2.5	96299
3.0	399042
3.5	240378
4.0	531498
4.5	169475
5.0	296463

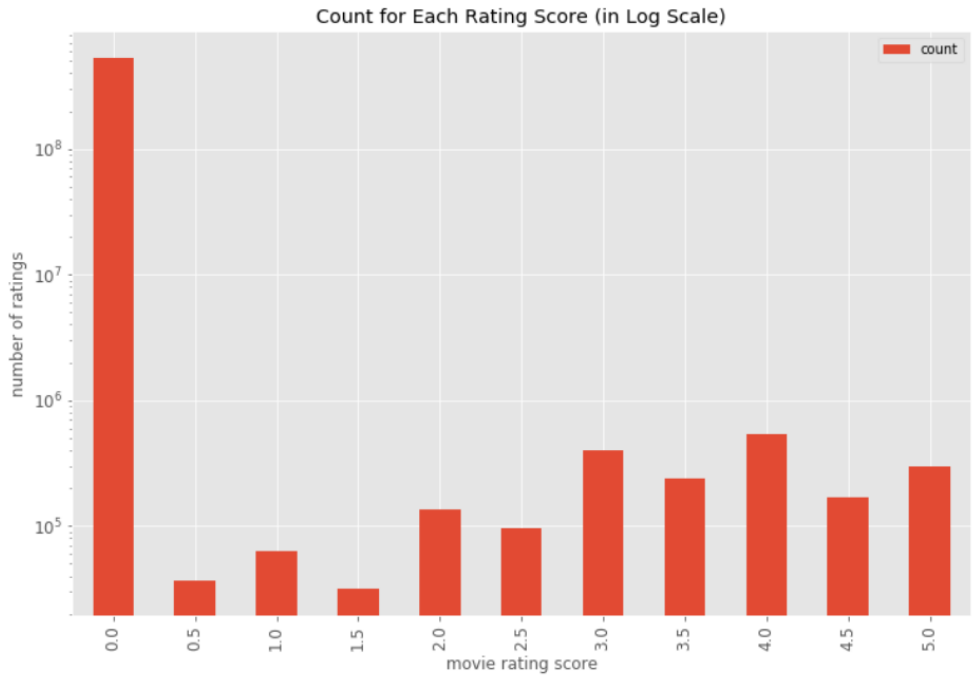
In [14]:
```
#normalise log to visualise easier see reference 1 and 8 in report
import numpy as np
df_ratings_cnt['log_count'] = np.log(df_ratings_cnt['count'])
df_ratings_cnt
```

Out[14]:

	count	log_count
0.0	531797210	20.091773
0.5	37006	10.518835
1.0	63892	11.064949
1.5	31587	10.360501
2.0	134360	11.808278
2.5	96299	11.475213
3.0	399042	12.896822
3.5	240378	12.389968
4.0	531498	13.183455
4.5	169475	12.040461
5.0	296463	12.599678

```
ax.set_ylabel( number of ratings )
```

Out[15]: Text(0, 0.5, 'number of ratings')



In [16]:
```
#get the ratings and number of ratings for each movie
df_movies_cnt = pd.DataFrame(df_ratings.groupby('movieId').size(), columns=['count'])
df_movies_cnt.head()
```

Out[16]:

movieId	count
1	4923
2	1975
3	1188
4	242
5	1138

In [18]:
```
# get the ratings from users
df_users_cnt = pd.DataFrame(df_ratings_drop_movies.groupby('userId').size(), columns=['count'])
df_users_cnt.head()
```

Out[18]:

userId	count
1	16
2	15
3	8
4	714
5	71

In [20]:
```
# create movie matrix, see reference 8 in report
movie_user_mat = df_ratings_drop_users.pivot(index='movieId', columns='userId', values='rating').fillna
#map movie titles to images
movie_to_idx = {
    movie: i for i, movie in
    enumerate(list(df_movies.set_index('movieId').loc[movie_user_mat.index].title))
}
# transform the matrix into a scipy, see reference 9 in the report
movie_user_mat_sparse = csr_matrix(movie_user_mat.values)
```

In [21]:
```
movie_user_mat_sparse
```

Out[21]:
```
<4802x7757 sparse matrix of type '<class 'numpy.float32'>'
        with 1611936 stored elements in Compressed Sparse Row format>
```

In [22]:
```
# define model see reference 7, 6 and 9 in the report
model_knn = NearestNeighbors(metric='cosine', algorithm='brute', n_neighbors=20, n_jobs=-1)
# fit
model_knn.fit(movie_user_mat_sparse)
```

Out[22]:
```
NearestNeighbors(algorithm='brute', leaf_size=30, metric='cosine',
                 metric_params=None, n_jobs=-1, n_neighbors=20, p=2,
                 radius=1.0)
```

In [26]:
```
user_input = input('Please enter a movie')

make_recommendation(
    model_knn=model_knn,
    data=movie_user_mat_sparse,
    fav_movie=user_input,
    mapper=movie_to_idx,
    n_recommendations=10)
```

```
Please enter a movieSpider Man
You have input movie: Spider Man
Found possible matches in our database: ['Spider-Man (2002)', 'Spider-Man 3 (2007)', 'Spider-Man 2 (2
004)', 'Spider (2002)']

Recommendation system start to make inference
......

Recommendations for Spider Man:
1: Lord of the Rings: The Return of the King, The (2003), with distance of 0.3985353708267212
2: Ocean's Eleven (2001), with distance of 0.389157772064209
3: Minority Report (2002), with distance of 0.38623547554016113
4: Lord of the Rings: The Two Towers, The (2002), with distance of 0.3744795322418213
5: Shrek (2001), with distance of 0.37224310636520386
6: Pirates of the Caribbean: The Curse of the Black Pearl (2003), with distance of 0.3719447255134582
5
7: X2: X-Men United (2003), with distance of 0.37088292837142944
8: Lord of the Rings: The Fellowship of the Ring, The (2001), with distance of 0.3658738136291504
9: X-Men (2000), with distance of 0.3391755223274231
10: Spider-Man 2 (2004), with distance of 0.28664588928222656
```