In [2]:
```
import pandas as pd
import numpy as np
from scipy.sparse import csr_matrix
from sklearn.neighbors import NearestNeighbors
import matplotlib.pyplot as plt
import seaborn as sns
movies = pd.read_csv("E:\Project\Dev Data set\ml-latest-small\Data set/movies.csv")
ratings = pd.read_csv("E:\Project\Dev Data set\ml-latest-small\Data set2/ratings.csv")
```

In [8]:
```
movies.head()
```

Out[8]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

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

Out[9]:

	userId	movieId	rating	timestamp
0	1	31	2.5	1260759144
1	1	1029	3.0	1260759179
2	1	1061	3.0	1260759182
3	1	1129	2.0	1260759185
4	1	1172	4.0	1260759205

In [10]:
```
final_dataset = ratings.pivot(index='movieId',columns='userId',values='rating')
final_dataset.head()
```

Out[10]:

userId	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**...**	**662**	**663**	**664**	**665**	**666**
movieId																
1	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	4.0	NaN	...	NaN	4.0	3.5	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	5.0	NaN	NaN	3.0	NaN
3	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	3.0	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	3.0	NaN

5 rows × 671 columns

In [11]:
```
final_dataset.fillna(0,inplace=True)
final_dataset.head()
```

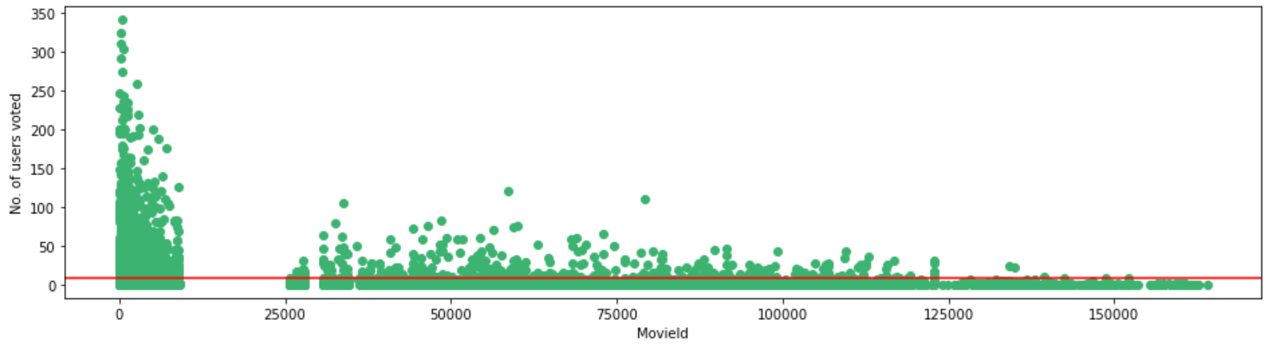
Out[11]:

userId	1	2	3	4	5	6	7	8	9	10	...	662	663	664	665	666	667	668	669
movieId																			
1	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	4.0	0.0	...	0.0	4.0	3.5	0.0	0.0	0.0	0.0	0.0
2	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	3.0	0.0	0.0	0.0	0.0
3	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	3.0	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
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	3.0	0.0	0.0	0.0	0.0

5 rows × 671 columns

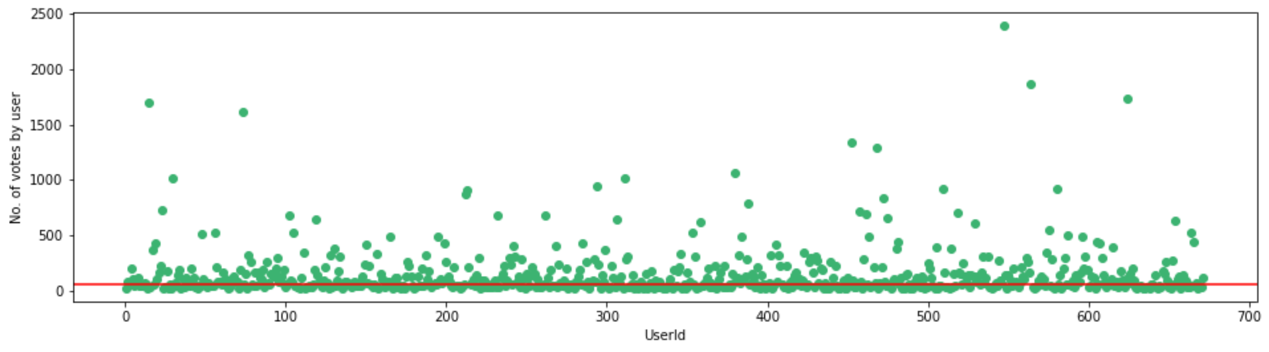
In [12]:
```
no_user_voted = ratings.groupby('movieId')['rating'].agg('count')
no_movies_voted = ratings.groupby('userId')['rating'].agg('count')
```

In [13]:
```
f,ax = plt.subplots(1,1,figsize=(16,4))
# ratings['rating'].plot(kind='hist')
plt.scatter(no_user_voted.index,no_user_voted,color='mediumseagreen')
plt.axhline(y=10,color='r')
plt.xlabel('MovieId')
plt.ylabel('No. of users voted')
plt.show()
```



In [14]:
```
final_dataset = final_dataset.loc[no_user_voted[no_user_voted > 10].index,:]
```

In [15]:
```
f,ax = plt.subplots(1,1,figsize=(16,4))
plt.scatter(no_movies_voted.index,no_movies_voted,color='mediumseagreen')
plt.axhline(y=50,color='r')
plt.xlabel('UserId')
plt.ylabel('No. of votes by user')
plt.show()
```



In [16]:
```
final_dataset=final_dataset.loc[:,no_movies_voted[no_movies_voted > 50].index]
final_dataset
```

Out[16]:

userId	2	3	4	5	7	8	12	13	15	17	...	655	656	658	659	660	662	664	665
movieId																			
1	0.0	0.0	0.0	0.0	3.0	0.0	0.0	5.0	2.0	0.0	...	0.0	0.0	0.0	0.0	2.5	0.0	3.5	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	...	4.0	0.0	0.0	0.0	0.0	5.0	0.0	3.0
3	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	3.0
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
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.5	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
122900	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	4.5	0.0
122904	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	0.0	0.0	0.0	0.0
134130	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.5	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
134853	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	0.0	0.0	0.0	0.0
139385	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.5	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

2083 rows × 421 columns

In [17]:
```
sample = np.array([[0,0,3,0,0],[4,0,0,0,2],[0,0,0,0,1]])
sparsity = 1.0 - ( np.count_nonzero(sample) / float(sample.size) )
print(sparsity)
```

```
0.7333333333333334
```

In [18]:
```
csr_sample = csr_matrix(sample)
print(csr_sample)
```

```
  (0, 2)	3
  (1, 0)	4
  (1, 4)	2
  (2, 4)	1
```

In [19]:
```
csr_data = csr_matrix(final_dataset.values)
final_dataset.reset_index(inplace=True)
```

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

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

Out[20]:
```
NearestNeighbors(algorithm='brute', metric='cosine', n_jobs=-1, n_neighbors=20)
```

In [21]:
```
def get_movie_recommendation(movie_name):
    n_movies_to_reccomend = 10
    movie_list = movies[movies['title'].str.contains(movie_name)]
    if len(movie_list):
        movie_idx= movie_list.iloc[0]['movieId']
        movie_idx = final_dataset[final_dataset['movieId'] == movie_idx].index[0]
        distances , indices = knn.kneighbors(csr_data[movie_idx],n_neighbors=n_movies_t
        rec_movie_indices = sorted(list(zip(indices.squeeze().tolist(),distances.squeez
        recommend_frame = []
        for val in rec_movie_indices:
            movie_idx = final_dataset.iloc[val[0]]['movieId']
            idx = movies[movies['movieId'] == movie_idx].index
            recommend_frame.append({'Title':movies.iloc[idx]['title'].values[0],'Distan
        df = pd.DataFrame(recommend_frame,index=range(1,n_movies_to_reccomend+1))
        return df
    else:
        return "No movies found. Please check your input"
```

In [38]:
```
get_movie_recommendation('Inception')
```

Out[38]:

	Title	Distance
1	Sherlock Holmes (2009)	0.410889
2	WALL·E (2008)	0.399529
3	Social Network, The (2010)	0.395383
4	Iron Man (2008)	0.385641
5	Shutter Island (2010)	0.382552
6	Dark Knight Rises, The (2012)	0.374756
7	Inglourious Basterds (2009)	0.361543
8	District 9 (2009)	0.343931
9	Avatar (2009)	0.302219
10	Dark Knight, The (2008)	0.283389