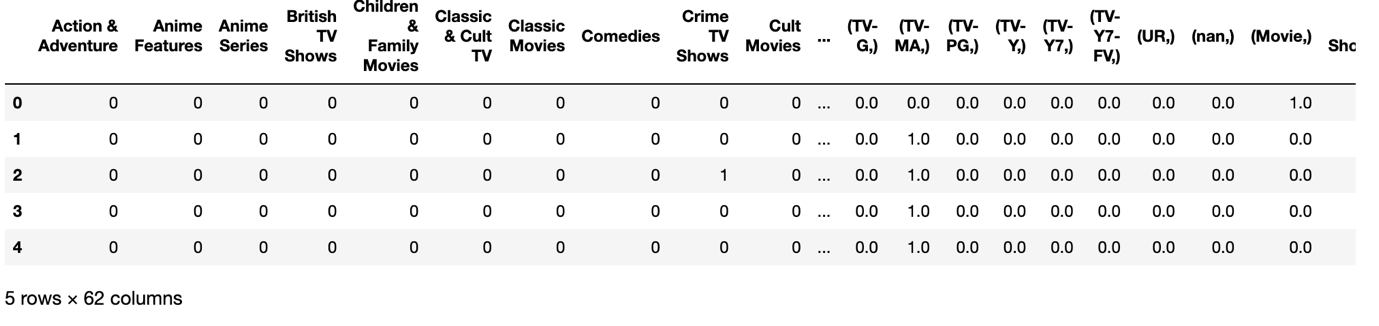
Using K\_means clustering with some techniques to build recommender system and clustering analysis

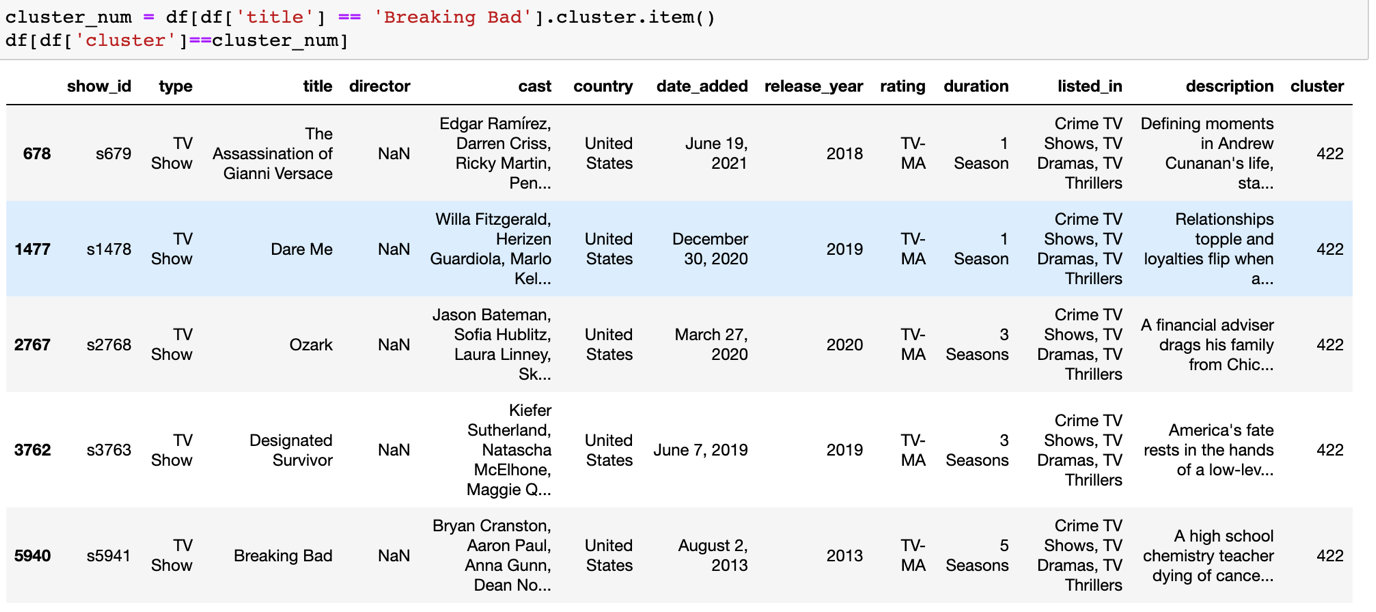
Recommender system:

In this section, I will use type (Movie/TVShow), country(country of film), rating(type of audience of film), listed\_in(type of film) to build recommender system.

* I used one hot encoder to encode the movie, tv-show and also encode rating column.
* Since a film has multiple countries and also multiple genre(listed\_in), so I used MultiLabelBinarizer() to encode that column. It is the same of one hot encode but each row will have >=1 value ‘1’. The result should look like this.



Then I choose n\_cluster=500 to cluster these films. Here is model recommend movie result when given movie ‘The Breaking Bad’.



The movie is belong to Crime TV\_show, Drama,… so the model will tend cluster movie that belong to these genre, there are other factor that contribute to this such as description(family, teacher, drugs, America,..).

Using K\_Means to group the main genre of movie.

Technical report:

* In order to do this, before stepping in k-means clustering. I had to apply some techniques to preprocessing the data (text in this case).
* After fill nan value with ‘unknown’. Then I concatenated the feature director, cast, country, listed-in and description column into a new data frame.
* Then I used some common technique in processing text data like remove punctuation like comma(,), full stop (.), colon(: ), semi-colon(;), question mark(?) and lower-case the text.
* After that I transform Unicode to Ascii to remove special symbol like:

“oración que está en español” into

“oracion que esta en espanol”

* After cleaning the text, I tokenized the text using WordNetLemmatizer() from sklearn. The purpose of this process is splitting each work in the text into token to remove the word that from StopWord(common word in English, such as the, an, a, …).
* Finally, I join the list into text.

\*All of this is combined in a function call clean\_text.

Next, I applied the TF\_IDF vectorizer to turn text in each row to vector in order to fit into K\_Means cluster.

The output of TF\_IDF is a matrix of size (num\_examples, features). Each column represent a word in the documents. The value of the matrix represent the TF\_IDF score of a word in given row(document).

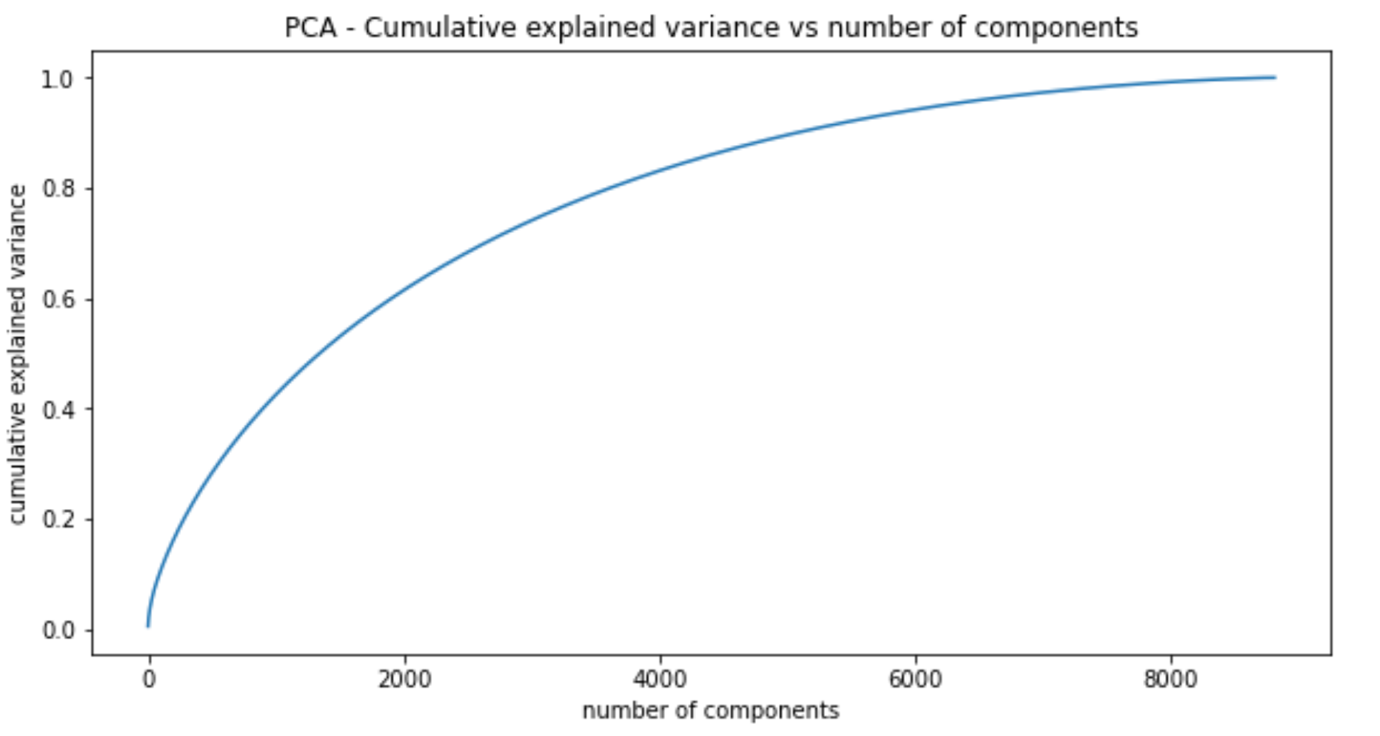
The formula of TF\_IDF is, TF-IDF(t, d) = TF(t, d) \* IDF(t)

The TF-IDF score gives higher weight to words that are frequent within a specific document but rare in the entire document collection.

I choose 20,000 words to be feature.

The result if the matrix in this case is (8807, 20000)

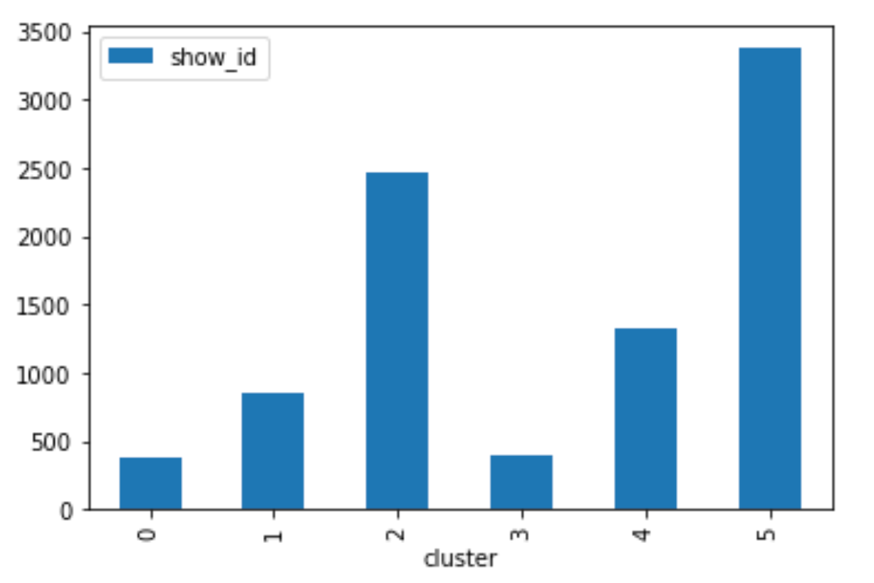
Next, I applied PCA to reduce the dimensionality of the matrix. To choose the perfect number of feature, I fit for all 1 -> 20000 feature and then collect their explained variance ratio. It means how many % of information given #nth feature captured in the total of 100%.



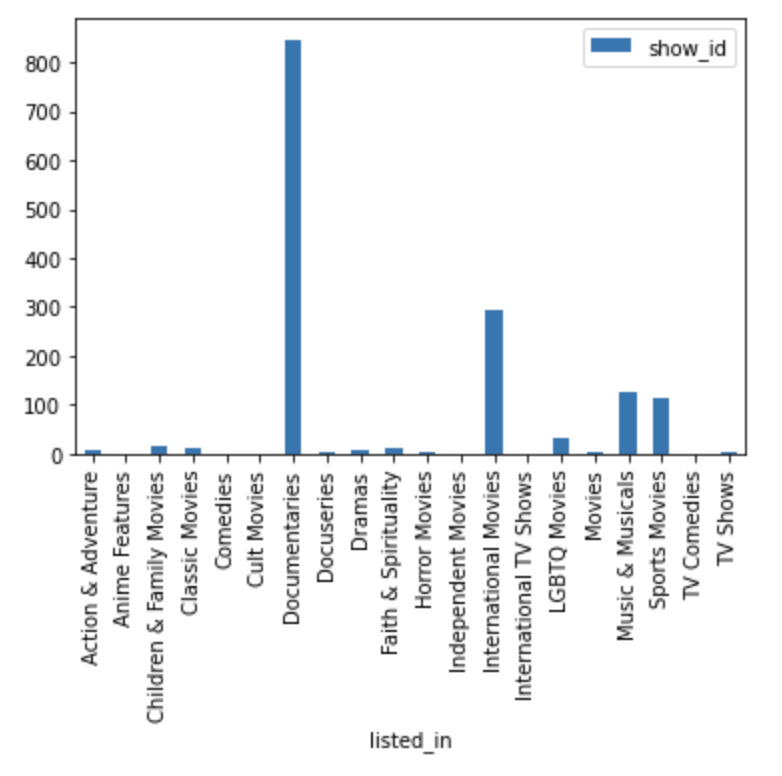
As you can see, 80% information captured with n\_feature =4000. So I chooses 4000 and then reduced matrix.

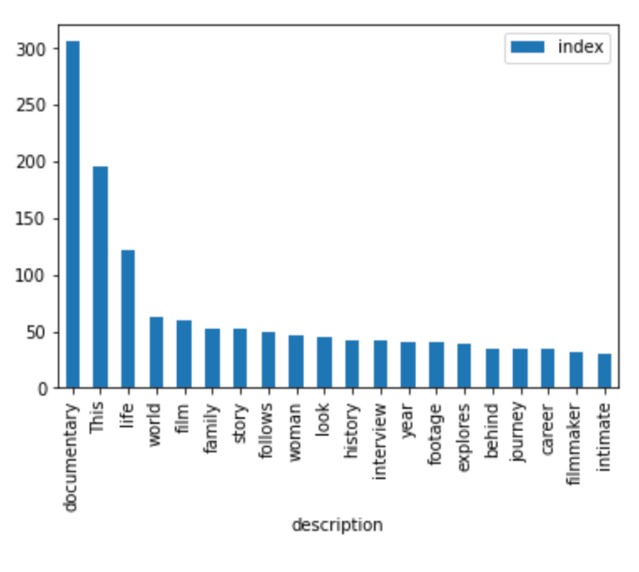
Finally, I would use knuckle test and silhouette score to choose the best cluster. After all, the perfect cluster is n=6.

The number of rows that belong to each cluster.

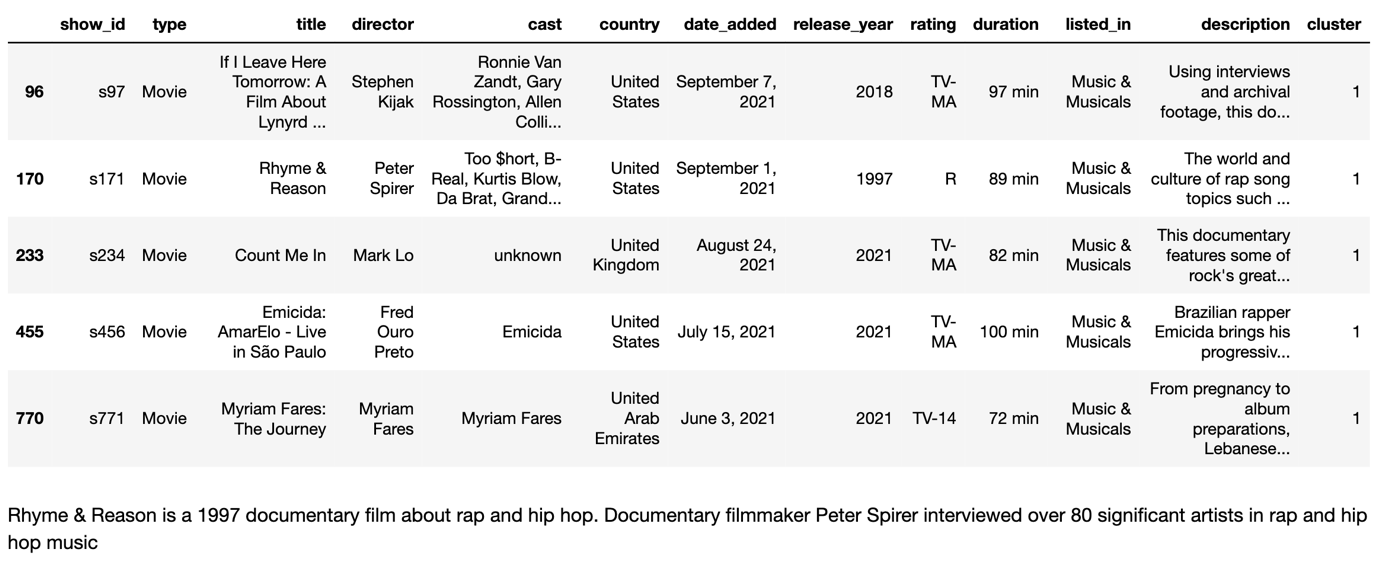


In this section, I will use K-means clustering to cluster the main genre in the dataset, such as Documentary, International movie, international tv\_shows. From this cluster, we can see which other minority genres usually go with majority genre.

As you can see, In the 1st cluster, where most films are documentary. Music and Sport are two type minority genre go with documentary. Let see the most frequent words in description.

From description, we can see that documentary talk about the life, career and family of some popular singer or footballer that impact the ‘world’. We also know how they make these film, including footage, explores or interview.

Here is some rows in cluster 1:



The content of documentary is exactly what I expected, life, music, interviews,..

For other cluster, please read my notebook.