What we have done in this module

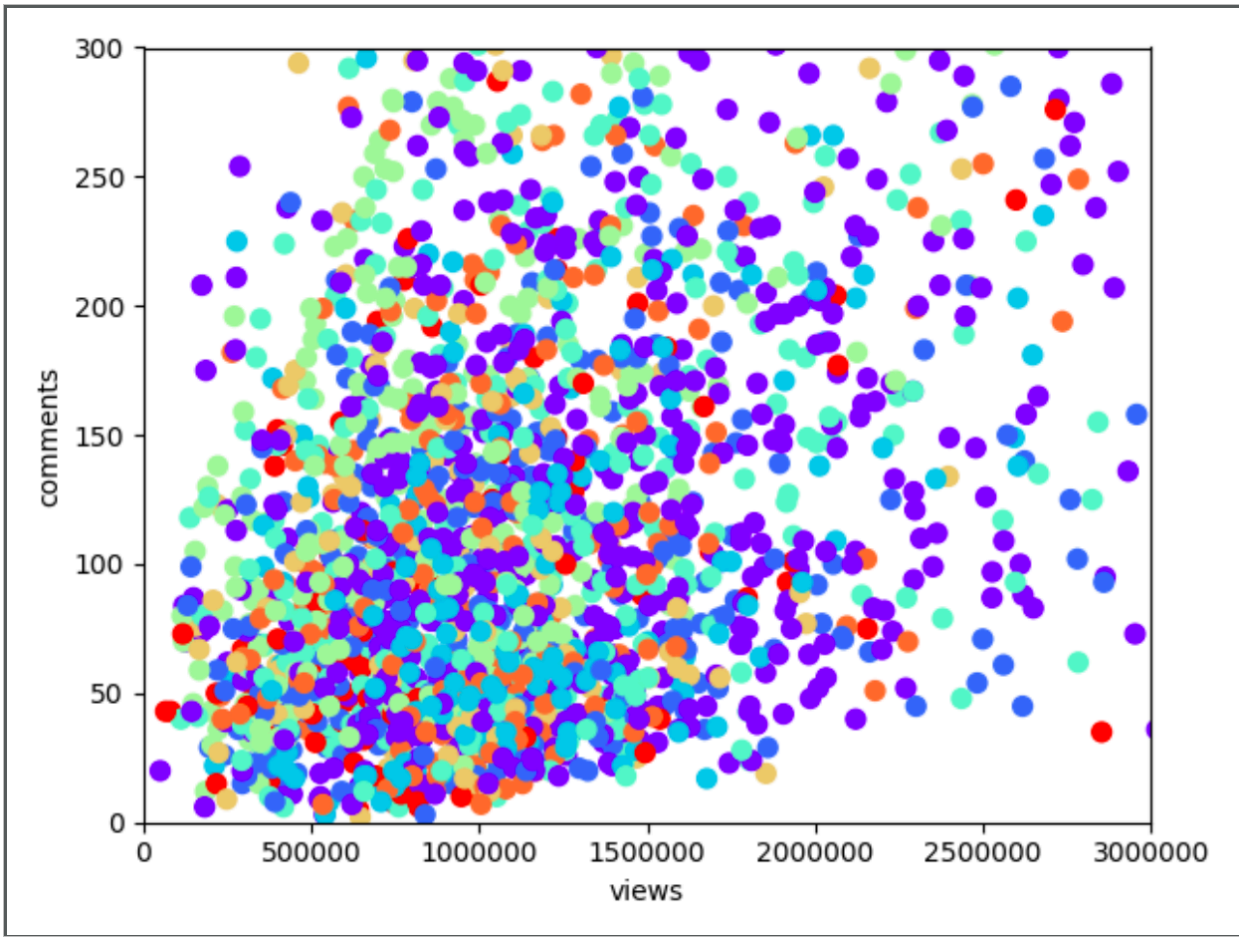
1. GetRecommendationModel.py

We calculate the TFIDF distance of the label corresponding to each video (the essence is to calculate the text difference between them), and get the TFIDF matrix, which is used as the metric of the K-Means algorithm, and the matrix is sent to the algorithm training, we will get a model.

```
tfidf = TfidfVectorizer(max_df = 0.8, stop_words='english')
movies['tags'] = movies['tags'].fillna('')
tfidf_matrix = tfidf.fit_transform(movies['tags'])
```

2. GetRecommendationReadFromModel

In this file, we can get a clustering result by reading the model stored in the previous file and display it on the scatter plot. The scatter plot is based on the amount of play and the amount of comments, each color representing a different clustering category (they are clustered by subject).



⚠️ **Difficulties：**

1. Our database lacks digital information, and more is textual information. Therefore, it is difficult to perform quantitative analysis.
2. As can be seen from the figure, the type and the amount of playback, the amount of commentary does not have much to do with the same type of playback, we need to conduct further correlation analysis.
3. The K-Means analysis based on the similarity of the label text is not very satisfactory, we may need to consider other algorithms.

3. recommendation.py

For the time being, the clustering algorithm is not considered. In this file, we can only get the ten videos that are closest to a speech by calculating the TF-IDF distance, thus playing the role of 'recommendation'. We are considering whether we need to make cluster recommendations.