Approach

As the prilimary step I have added the **YouTube Data API** v3 to my account and enabled it , according to the guidlines mentioned in google console.

I chose my categories to be DATA SCIENCE , COMPUTER NETWORKS AND WIRELESS COMMUNICATION.

Data cleaning

I have written the code by filtering tha query results according to the requirement of the assignment i.e., which columns to be kept and which are to be discarded.

Data Collection:

I have collected the data for 4 consecutive days and stored them in different CSV files so that it would be easy for me to compare between the trends.
Each day I have collected about 250 rows (which is 5 pages) in each category.
So, In total I have collected 750 rows per day .
Repeating the same process for 4 days gave me a total of 3,000 rows

Data manipulation:

So as to visualize the trends in likes count , dislikes count ,views count from the data and to rank them I have collected, I have took a differnce between likes count, dislikes count and views count of day4 and day1 data, by calling a merge function on the main data frame on videold, videoTitle, published date , duration and topic. What this basically does is that it takes the differnce if a particular row from day4 data matches with the mentioned parameters with a row in day1 data.

Now this is our final data which we will be working upon to create a playlist and to analyse the trends.

To get a better estimate I have also took the difference between the likescount_diff and the dislikescount_diff which gives me the net difference of likes count and dislikes count.

To make things simpler , I have grouped this main data frame into 3 groups based on the topic.

To scale down the values I have normalised the like_dislike_diff and viewcount_diff
st, $x_i = x_i / \text{sum}(x_i)$ (i goes from 0 to len data frame of each topic) .
Here , I have used sum in the denominator so that i would be getting the fraction of each entry wrt the total entries.i.e, to find the contribution of each entry [I have done this for all the 3 topics] .

Scoring Function:

Then I have given weights to the normalised like_dislike_diff and viewcount_diff values that goes into my scoring function. I have given a relatively higher weight to normalised viewcount_diff than that to normalised like_dislike_diff , because it so happens that people who might just watch the video and just leave without even giving their review on it . so considering this i have chose to give a bit higher weight to normalised viewcount_diff.

Calculating the scores:

I have calculated the scores for all the rows for each individual topic and sorted them in descending order.

Creating a playlist :

Now, for each topic i have taken out the videos from the sorted videos list above such that their sum of durations dosent exceed more than 15 hours.
doing this for each individual topic gives me a playlist of 15 hrs for each topic.

Analysing:

Now that we have everything in our hand I have plotted a graph between video Title of each respective playlist that we have created and their duration.