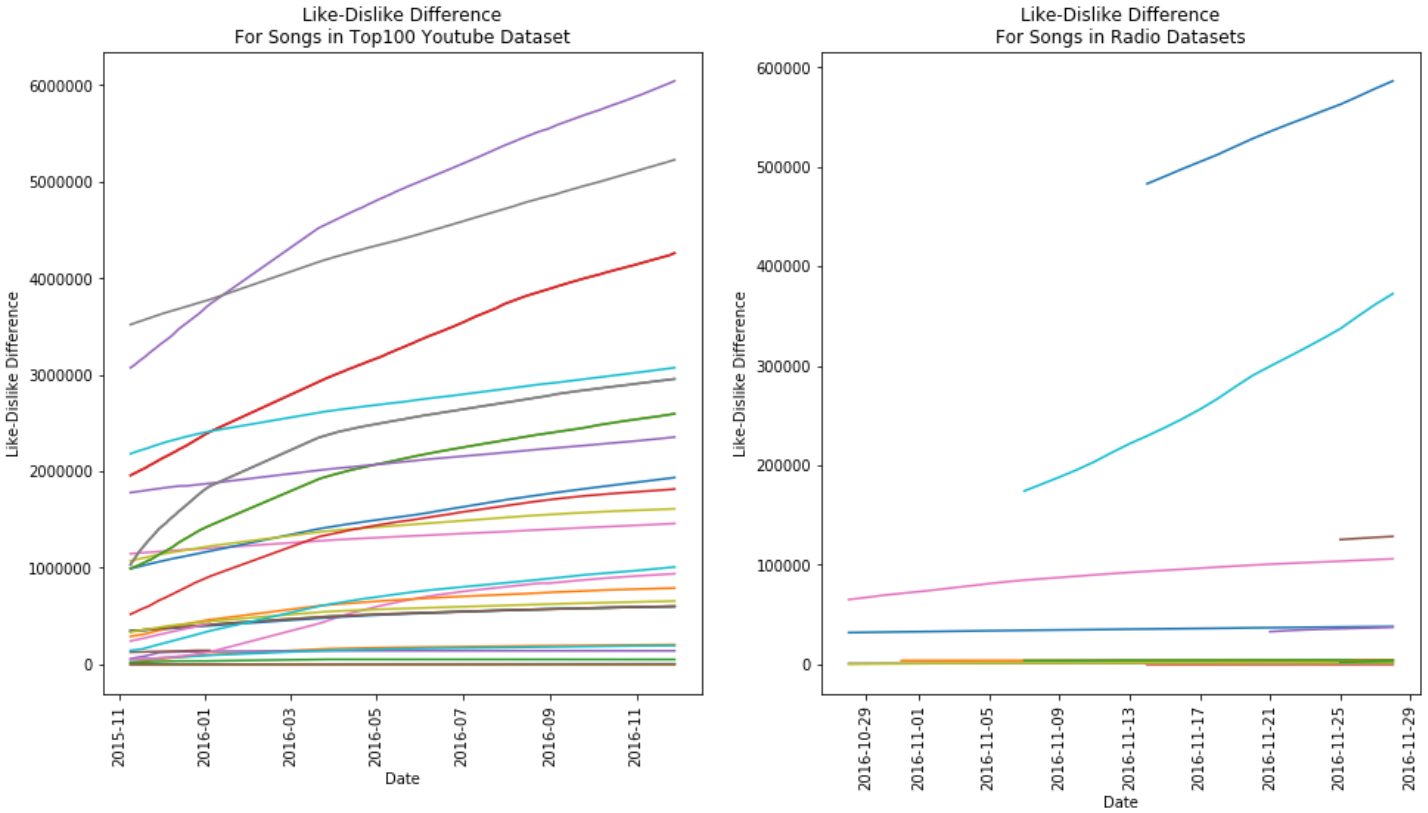
Web Science
Week 3-4 - Project Report

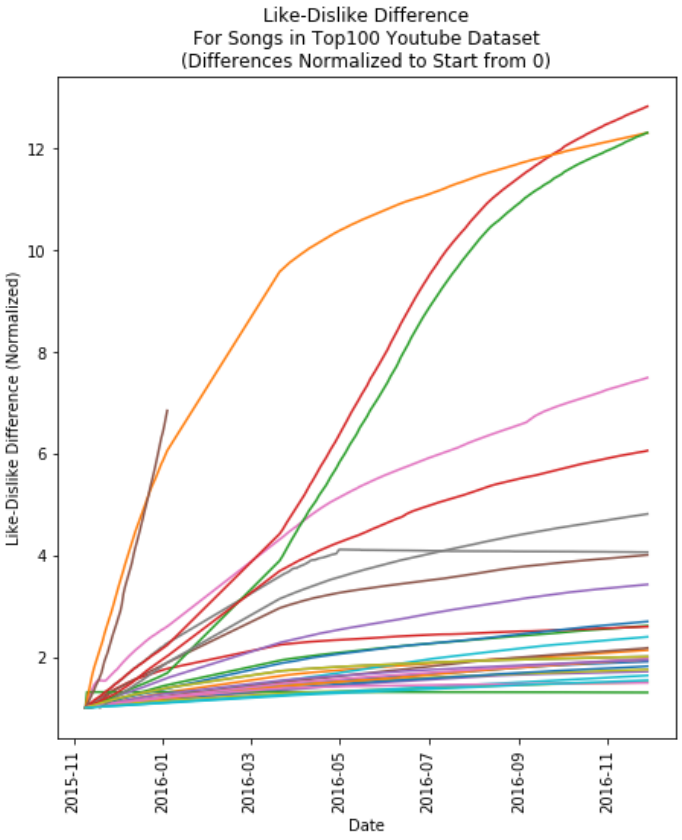
Group 34
Cahid Arda Öz - 2872293
Carlos de Bourbon de Parme – 2153874

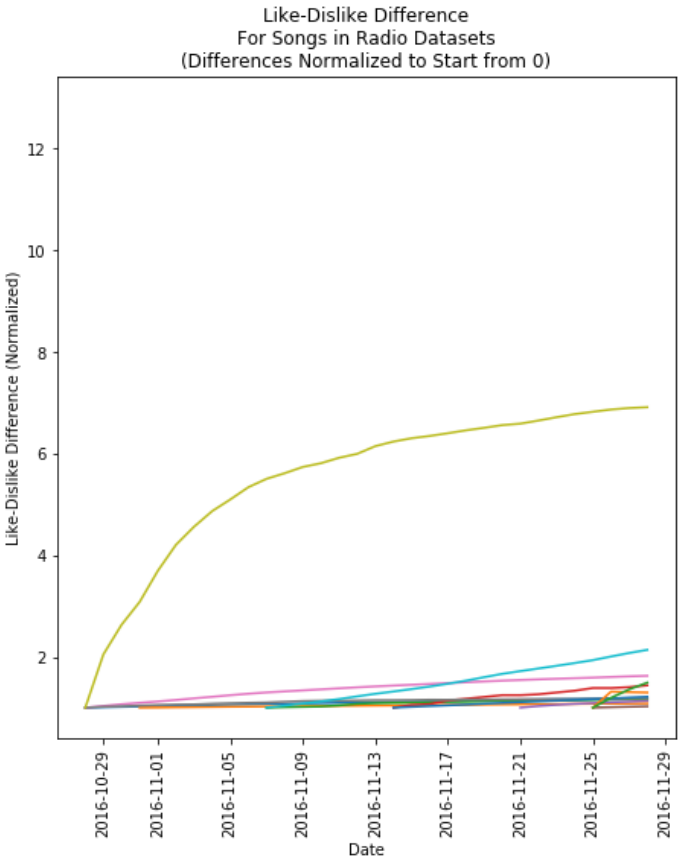
Assignment 1

To investigate whether cascading effects are prominent, we first compare 30 random songs which are in the top 100 youtube songs dataset and the songs in the radio hit datasets:



In the plots above, we see that most of the popular songs shown in the left graph are getting more and more likes while most of the non-popular songs in the right graph have a horizontal trajectory. To visualize this effect better, we wanted to normalize the like-dislike difference of the songs. To do this, we divided the series of data points corresponding to each song and divided the series by the first element in the series. This resulted with the following graphs:





Looking at the graphs above, we make the following observations: Compared to the songs in the radio datasets, songs in the Top100 Youtube dataset reach a higher number of like-dislike values compared to their initial values. This can be seen in the graphs above by looking at the highest ratios reached in the two graphs. Songs in the Top100 Youtube dataset reach values as high as 12 while songs in the radio dataset only reach 7. We also note that the song that reaches 7 in the radio dataset looks like an outlier since no other song has come close to it.

We see that the model in the book can not be applied here directly. This is because in order to use the model, we need to be able to observe the sequential responses of the users. In our data, we are only able to see decisions made over a day. Instead of individual responses, we can only observe aggregate responses.

Code

In the soluion of the assignment 1, we have wrote several helper methods. These are:

- get_files: returns a list of the data files in a data folder
- load_dict: loads a data file
- load_df: loads all of the files in a data folder to a dataframe
- get_song_dfs: gets a list of dataframes. Each dataframe represents a song

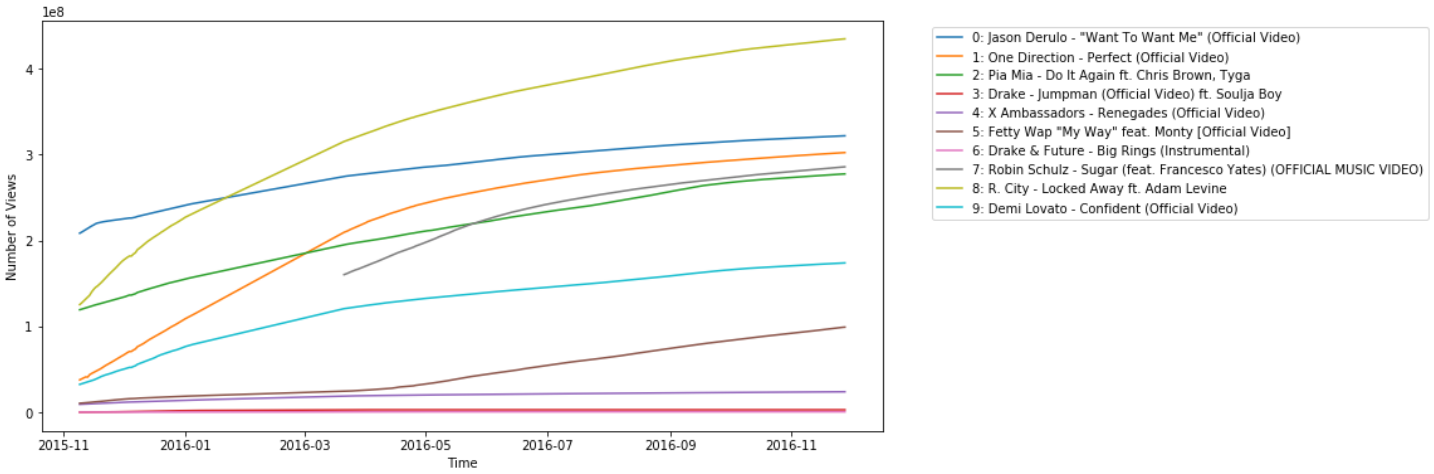
These helper methods are used to load data from Youtube file. Then dataframes are created for 10 songs and difference of likes and dislikes are calculated in each date. Then this difference is plotted for each song. Then another plot is generated by normalizing the songs by dividing the like-dislike difference series by the first day of the series, making each song's series start from 1.

Assignment 2

In this assignment, we investigate the network effects using the youtube dataset. We assume that without network effects we can expect that users visit a website with a certain frequency and view the song if it matches their taste. Hence, the expected number of views grows linearly in time.

In our investigation, we used the youtube_top100 dataset. We chose this dataset for this investigation because it contains data of the same 100 songs over a period of time. After loading the data, we also interpolate to fill the missing dates.

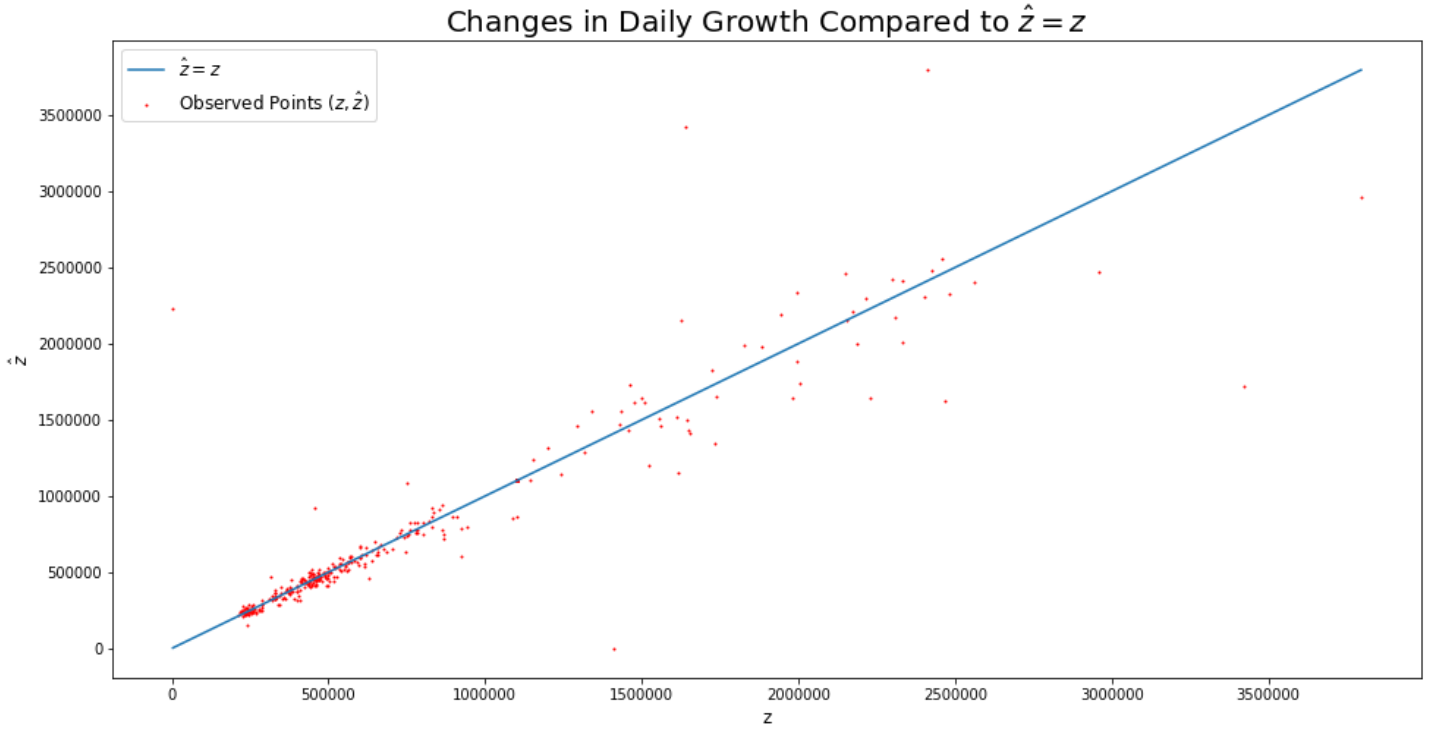
We start by selecting 10 songs from our dataset randomly and plotting the number of views over time:



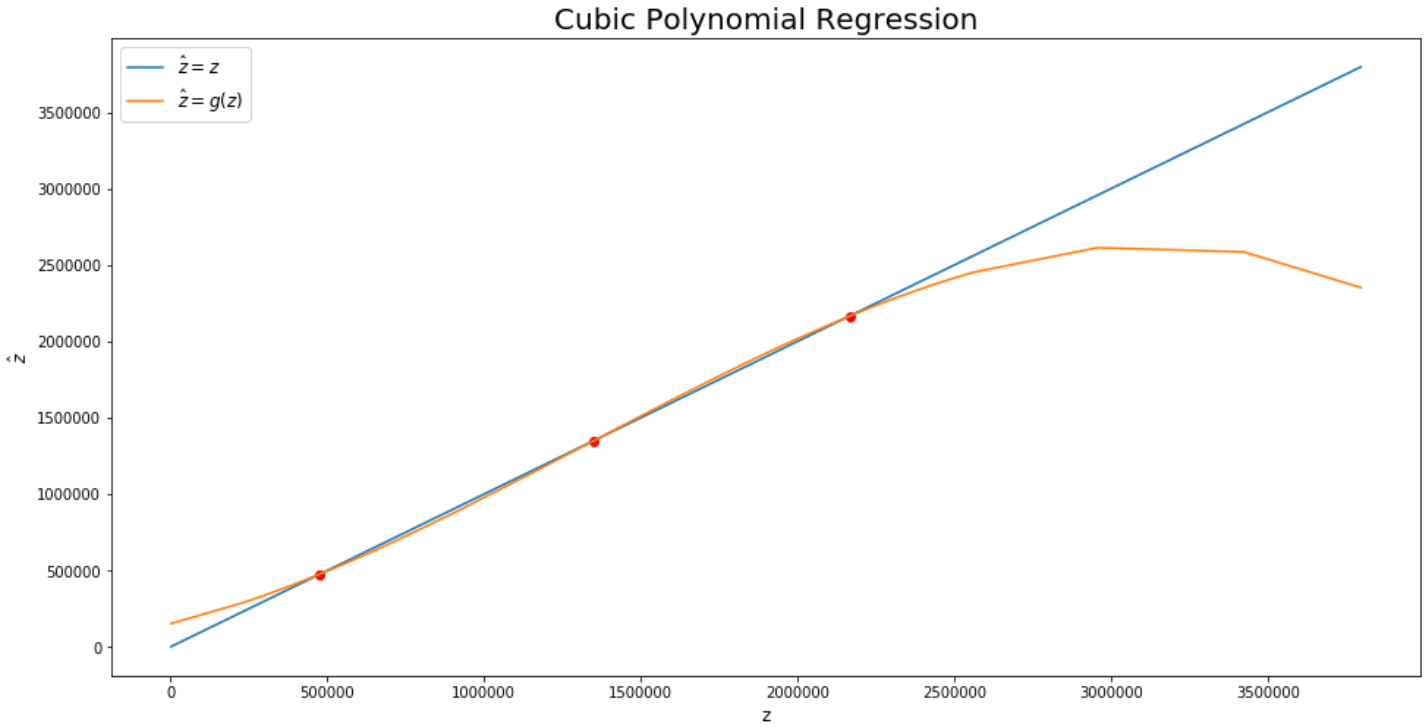
We see that for some of the songs, the increase in the number of views is not linear. Our assumption told us that without network effects, the number of views increases linearly. Since the increase in the number of views is not linear, we conclude that network effects are effective in our Youtube data.

When we compare our plot with Figure 17.4 from the book, we see that network effects can explain what we observe in our plot. Consider the song “R. City - Locked Away ft. Adam Levine” (8th song in our samples). Increase in view count gradually decreases over time. This suggests a downwards pressure. It is likely that the number of users is higher than the second equilibrium point and the number of users is gradually decreasing towards the equilibrium point.

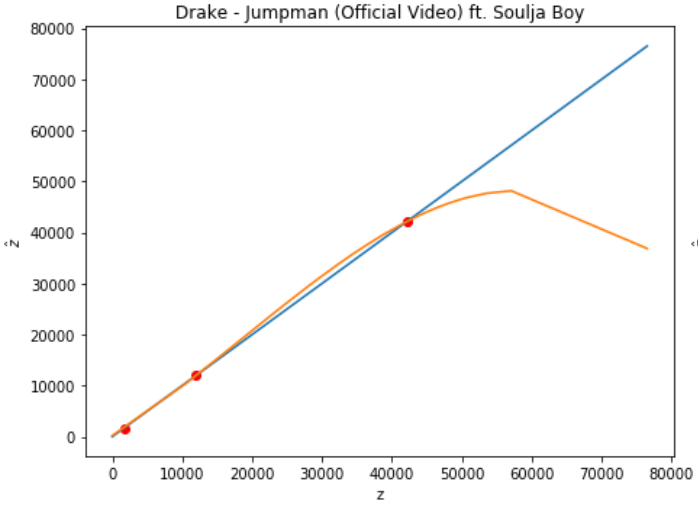
We wanted to visualize the network effect by plotting the same graph as Figure 17.5 for the songs in our dataset. We consider the daily increase in the view count to be the number of users. We look at the difference of view count between day *t* and *t+1* to calculate the number of users in day *t*. We then use this data to create the following graph. Each dot represents a day; x axis being the number of users on day *t* and y axis being the number of users on day *t+1*.

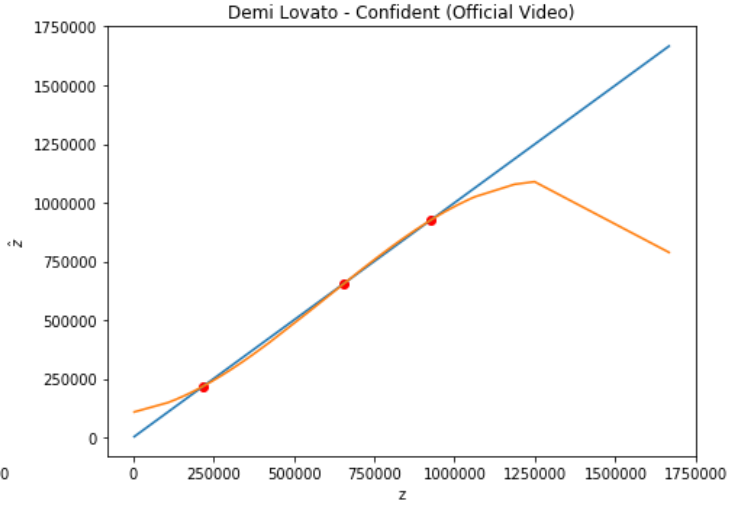


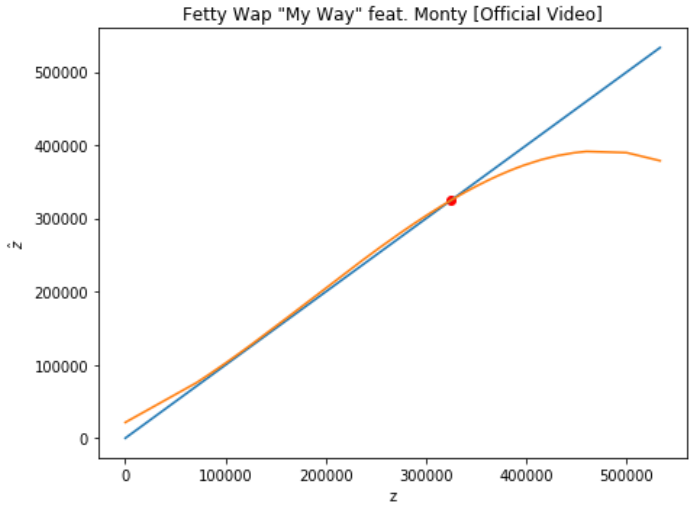
Next, we fit a regression model to our observed points. We used a cubic polynomial to model the data:

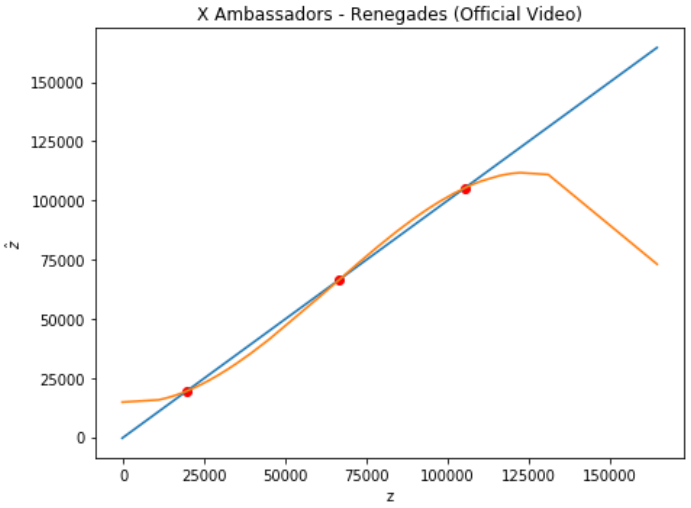


We see that in this example, there are three intersections forming three equilibriums. First and third equilibriums form a stable equilibrium while the second one in the middle is unstable. We then applied the same model to other songs to generate the following plots:









Looking at the graphs, we make the following observations:

- For z=0, the network benefit function f(z) has a positive value (f(0)>0) in most of the songs we observed. This is because the g(z) function is not zero around z=0, instead it has a positive value. Hence, the songs have value even if no one else is listening to them.
- Intrinsic interest function (r)
- When we considered the Price (p*) parameter, the song named 'Fetty Wap "My Way" feat. Monty [Official Video]' came to our attention. We only observed one place where the g(z) crossed $z = \hat{z}$. This is a similar case to Figure 17.12 in the book. *Price* of this song is relatively low compared to other songs and this song is able to gain more and more views until it reaches its stable equilibrium point

Code

We assume that the daily change viewCount corresponds to the number of viewers of a song. The number of viewers each day also changes from date to date. To calculate these, we wrote methods additional to the ones we had written in question 1. These methods are:

- get_view_growth: Used to calculate the increase in view growth each date. This increase represents the number of viewers
- get_z_hat_df: For a day t, let the number of viewers be z. Than the number of viewers next day is $\hat{z}$. This method is used to generate all the $(z, \hat{z})$ pairs.
- apply_p: This method is used to draw the fitted polynomial by applying the polynomial represented by its coefficients to a series of points

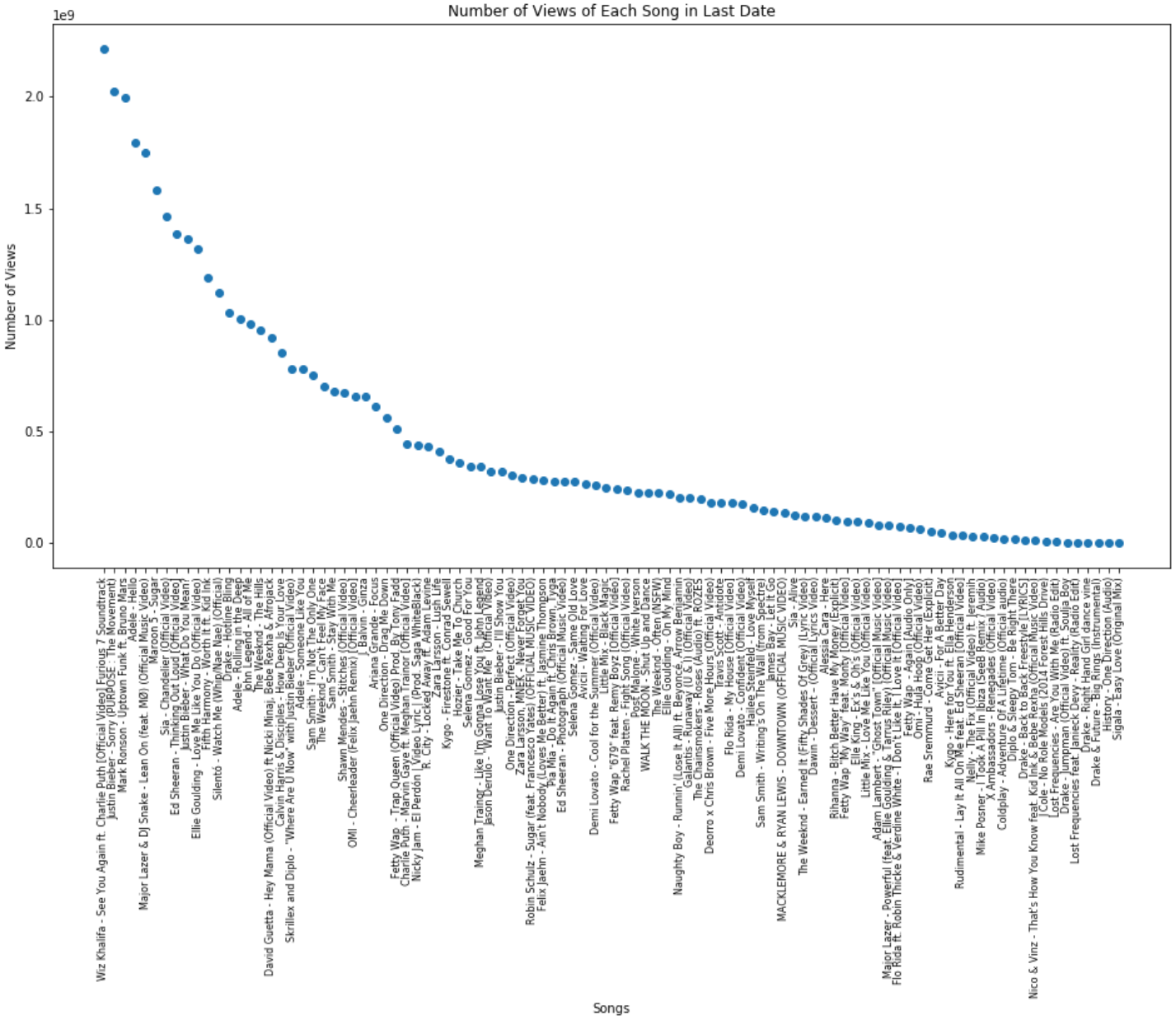
With the methods we wrote, we first load the Youtube data. Then we plot the view counts of some of the songs over time. We also plot the increase in view counts in a log scale because we will use it in question 3.

We then start working on the daily change in daily growth. We do this by generating a dataframe using get_z_hat_df method and plotting the resulting data points.

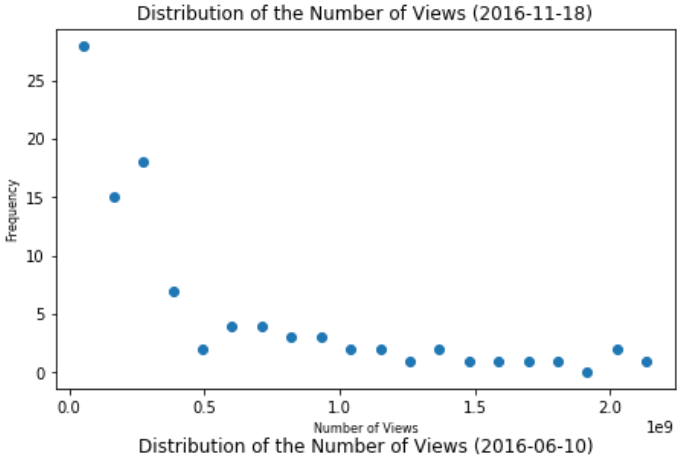
We then tried to join these data points. We first attempted to use Exponential Smoothing but the results were not satisfying. Then we fitted a cubic polynomial to our points. We were satisfied with the results and we decided to apply the same approach to 4 other songs.

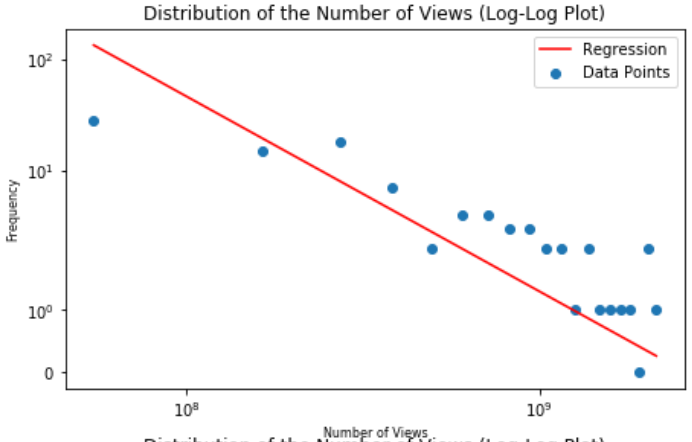
Assignment 3

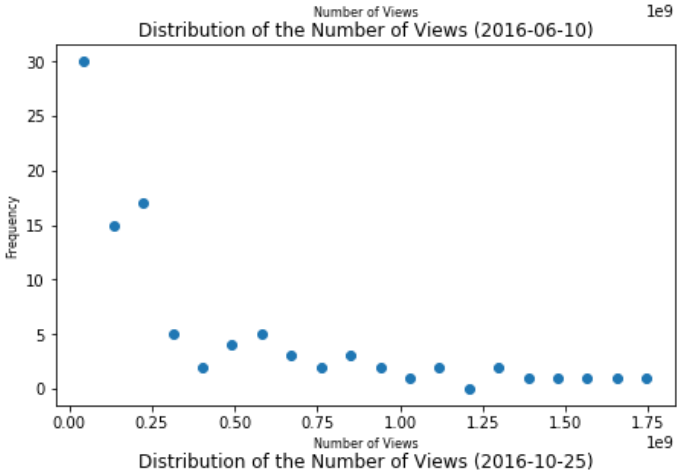
To investigate whether power laws play a role in our data, we first used the Youtube data to plot the distribution of the number of views in the last date in the dataset:

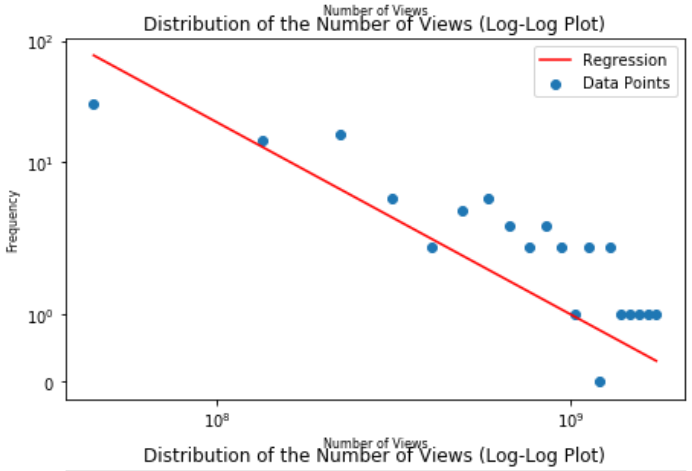


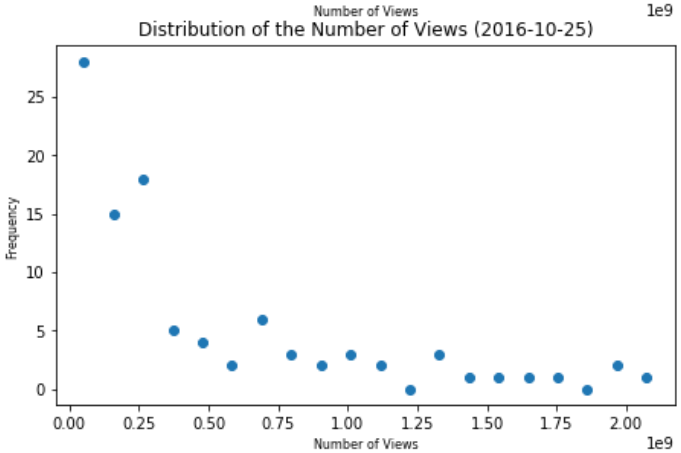
We see that there are a few songs with high number of views and a lot of songs with low number of views. This suggests that there may be a power-law relationship in our data. Next, we plot the distribution of the number of views in order to observe the power-law relationship better. In the graphs below, you can see the distribution plots. Each row corresponds to a randomly picked date from the Top100 Youtube dataset. In each row, left graph shows the distribution of the number of views with a normal scale and the right graph shows the same distribution in a log-log plot. We observe that the distributions are similar to the power-law graphs and the data points in the log-log plot form a line.

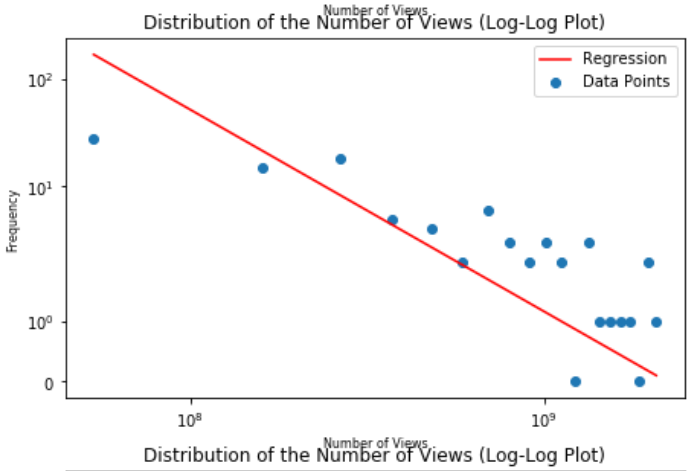


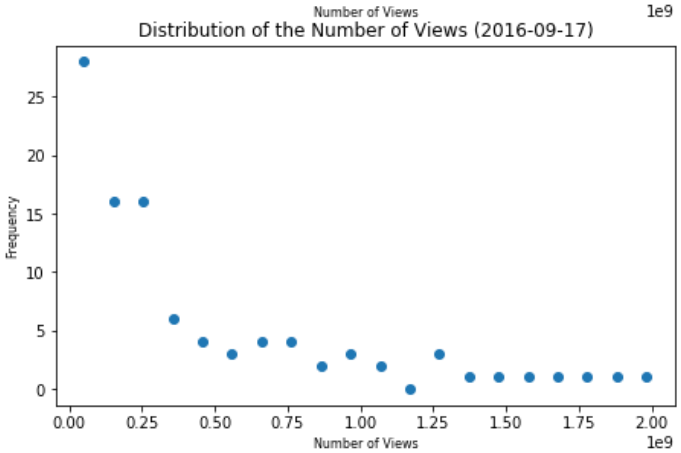


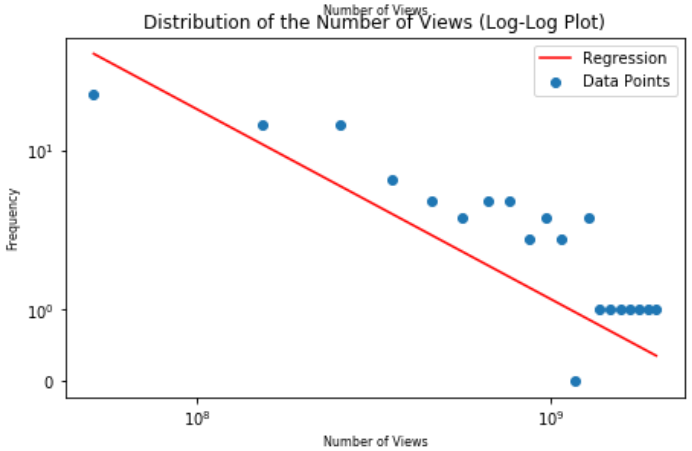












Next, we will show that the number of views grows exponentially in time when we assume that the number of views each day is proportional to the previous day. Let $v_d(n)$ represent daily views in day n and let $v_t(n)$ represent the number of total views until day n. Assume that $v_d(0) = k$, meaning that on day zero we have k views. In the beginning, we assumed that the number of views each day is proportional to the previous day. This means $v_d(n) = r * v_d(n-1)$.

Using our assumptions, we first show that $v_d(n) = r^n k$:

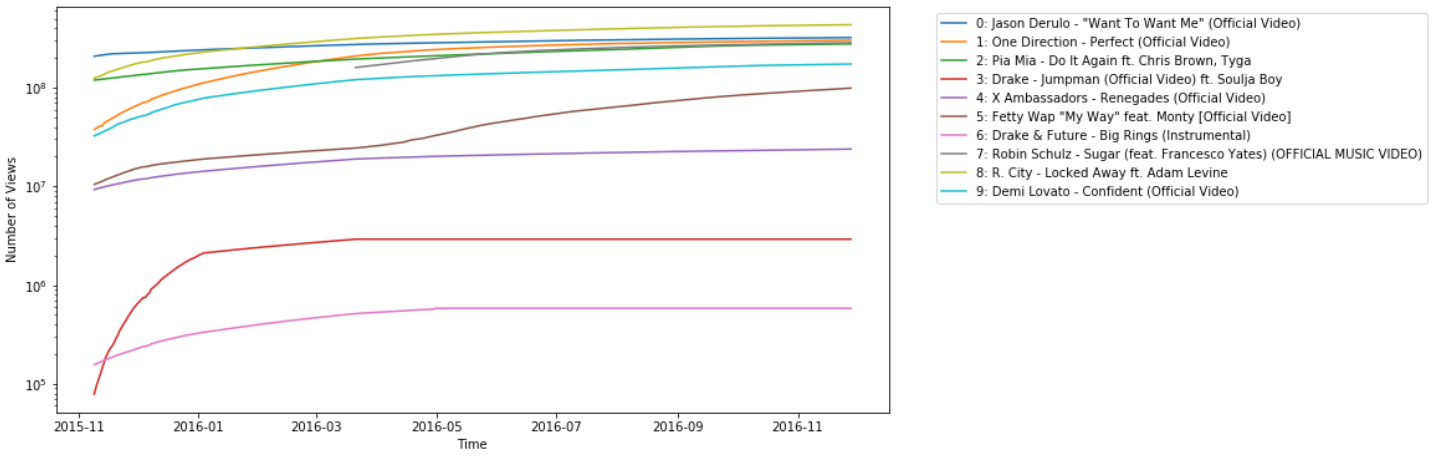
$$= v_d(0) = k$$
$$= v_d(1) = r * v_d(0) = rk$$
$$= v_d(2) = r * v_d(1) = r * rk = r^2 k$$
$$= \ldots$$
$$= v_d(n) = r * v_d(n-1) = r * r^{(n-1)} * k = r^n k$$

Next, we find the relationship between $v_t(n)$, r and k:

$$= v_t(0) = v_d(0) = k$$
$$= v_t(1) = v_d(0) + v_d(1) = k + rk = k(1 + r)$$
$$= v_t(2) = v_d(0) + v_d(1) + v_d(2) = k + rk + r^2 k = k(1 + r + r^2)$$
$$= \ldots$$
$$= v_t(n) = v_d(0) + \ldots + v_d(n) = k + \ldots + r^n k = k(1 + \ldots + r^n) = k(\frac{r^{n-1}-1}{r-1})$$

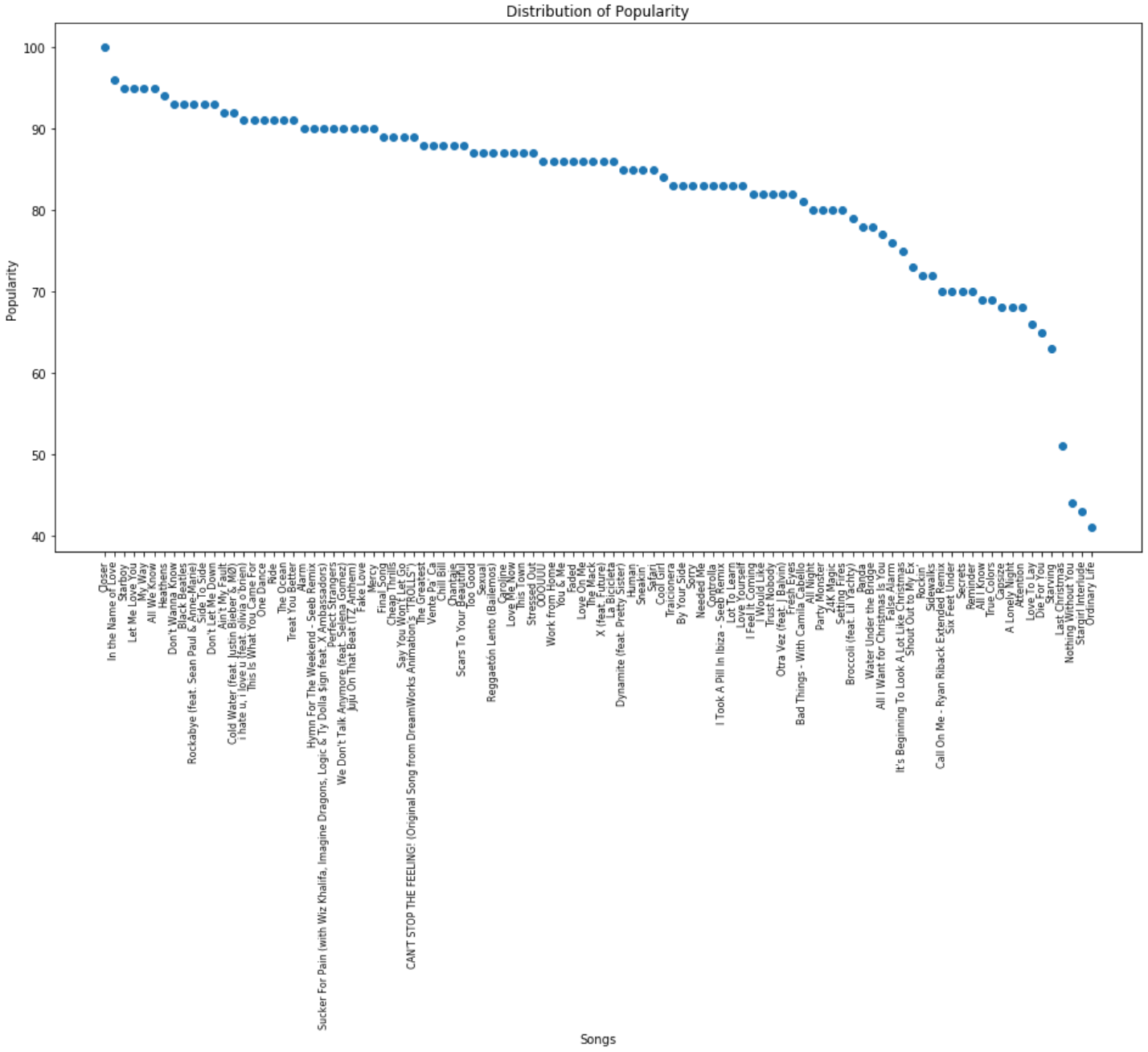
This tells us that $v_t(n)$ is an exponential function. Based on the assumption that the number of views each day is proportional to the previous day, the number of total views will grow exponentially in time.

When we compared our conclusion about $v_t(n)$ with the graphs we generated for question 2, we saw that our $v_t(n)$ the model didn't explain the increase in the number of total views over time. To confirm this, we plot the graph we generated in assignment 2 with y axis with a log scale:

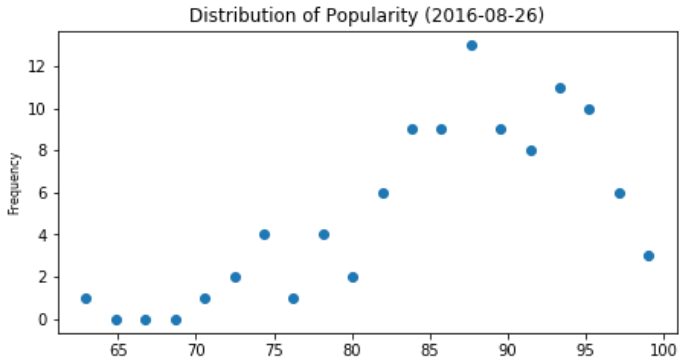


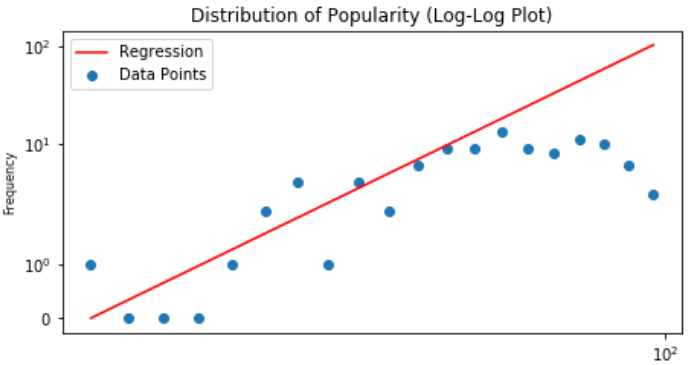
We see that songs don't have a steady exponential increase. Their views often start increasing in a non-exponential way and then this increase slowly fades away resulting in a more or less flat trajectory. The initial increase is not exponential because the increase doesn't form a line in the log scale. Therefore, we conclude that the assumption that the number of views each day is proportional to the previous day doesn't hold for our data.

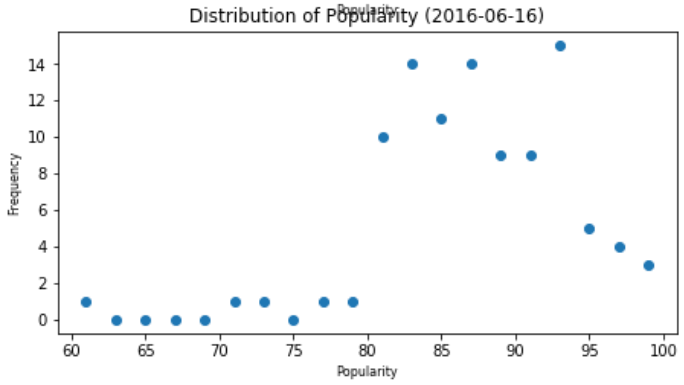
Last, we investigate the Spotify data to see if there is a power-law relationship. We look at the most popular songs from the last date and plot their popularies. Unlike the data in the Youtube dataset, this time we don't observe a power-law. There are a lot of songs with high popularity while there aren't many songs with low popularity:

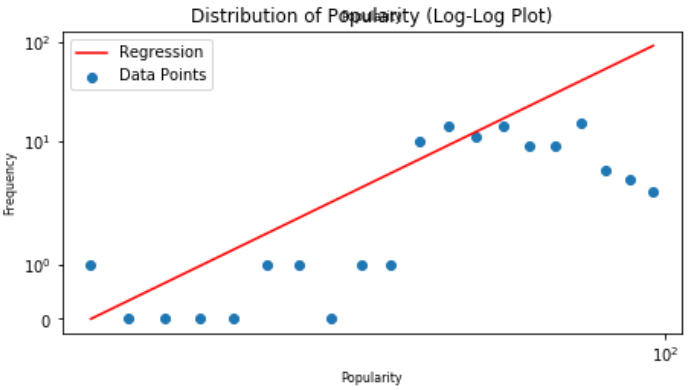


To investigate Spotify popularity further, we plot the distributions like we did for Youtube dataset. You can see the result down below:









We observe that the distributions look nothing like the power-law relationship we observed in Youtube dataset. This may be because the songs in the Spotify dataset are not randomly picked from all the songs uploaded to Spotify. All the songs in the Spotify dataset belong to a special subset, namely the top 100 songs. This sorted selection can be the cause of this result.

Code

In this question, we used the same helper methods we used in the previous questions.

We loaded the Youtube dataset and plotted the number of views against songs. We observed that there is potentially a power-law relationship. Next, we randomly picked 4 dates from our dataset.

For each date, we picked the rows that correspond to this date, found the distribution using *np.histogram* and plotted the distributions. Then we applied log function to both x and y coordinates of the distribution we generated and fitted a line to the resulting series. After we fitted a line, we drew this line and also plotted the distribution points with a log-log scale. In the end we have two plots for each date, both plotted in a single row of the subplots.

We then applied the same approach to the Spotify dataset.

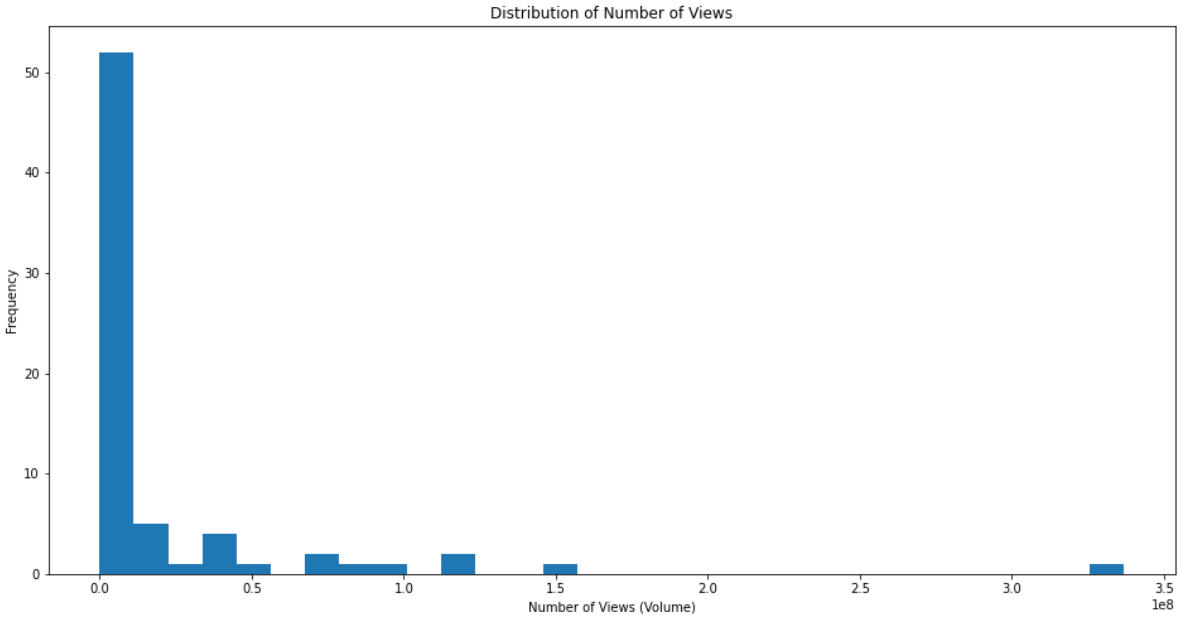
Assignment 4

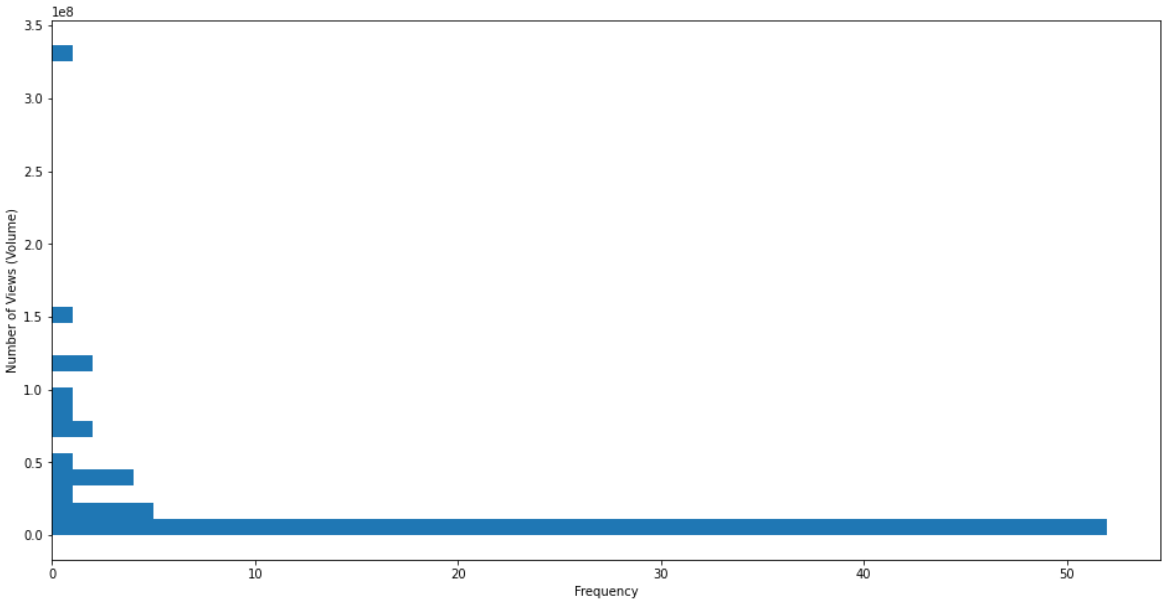
We believe that the model of information diffusion can be applied to our data. In the model of information diffusion, the rate of increase in the number of vertices depends on the shape of the network. This rate of increase can speed up or slow down depending on the shape of the network. When we look at the first graph we plotted for assignment 2, we see that the rate of increase slows down in some songs while it becomes faster in other songs over time.

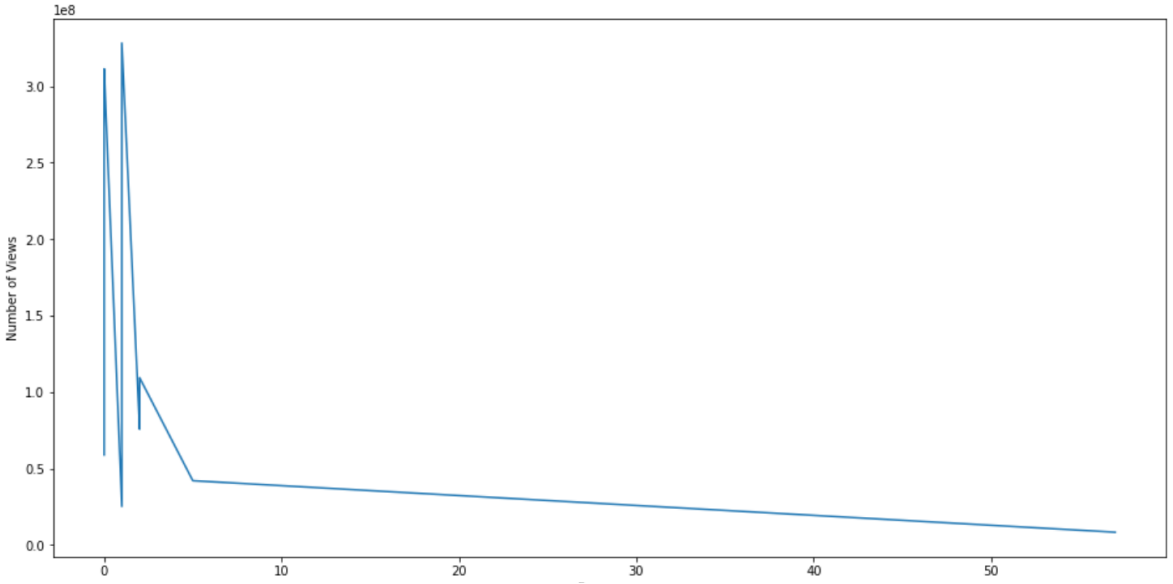
An hypothetical example of this can be that a new song from a less popular niche genre will take longer to reach the general viewers of youtube. Youtube will recommend this new song to people who enjoy this genre of music, if these people don't reference this song to others from outside this genre the song doesn’t get discovered by the broader audience.

Assignment 5

We queried the top 5 recommendations per song, and randomly picked the next song from those 5. We continued this process 100 Times. Unfortunately we had some difficulties with the youtube data api (other than the rate limit). We ended up with 71 songs and statistics. The songs were then grouped by views into 30 bins and plotted as histograms.
The first graph shows the frequency of videos on the Y axes, with views going from low to high on the X axes. The second graph the axes are filpt to conform to example figure 18.4 from the book. Which we also plotted as a line for completeness.







We can observe that that shape resembles that of a power law distribution, characterized by its long “neck” and “tail”.
As can be seen from both graphs, there is a long tail phenomenon, meaning that a lot of videos have a relatively low view count, but when summed up, are still responsible for a large number of total views. The bars of the graph are not all in descending size, the distribution is a bit noisy / random but the graph overall still looks similar to 18.4 example. This randomness is probably partly due to some missing data points (29 of the 100 samples could not load the views accounts), which would explain the lack of videos in certain bins. We expect that a larger sample of videos would result in a more even distribution with an overall similar shape.

Code

Due to restrictions we came across when using Youtube API, we retrieved the data in 2 steps:

- We created a list of video ids. Each succeeding id is in the recommendations of the video with the previous id. We saved the ids in a list
- We then iterated over the list to retrieve statistics for each video in the list. We build a dataframe with the data we received.

Using our dataframe, we plotted the distribution of the number of views to investigate the presence of power-law relationship.

Assignment 6

We believe that power-law relationship is a powerful tool when analyzing the youtube data. We observe a power law like distribution for youtube data in both assignment 3 and assignment 5, indicating that some songs have a strong market share. We believe it is partially due to signals, such as the like count or view counts. These signals help people decide which song to listen to, and help youtube know which songs to recommend. This supports the “richer get richer” phenomenon. Further proof for this effect is that when looking at the results for assignment 2, we do observe a more linear distribution for the spotify results, a platform where the popularity signal is not as prominent as it is on Youtube.

We believe that some models could have been more successful if we had the necessary data to use them. These data are as follows:

- Cascading: Data contained like and dislike counts based on dates. Being able to observe the change after each like or dislike would be more ideal for the model.
- Information Diffusion: In the case of information diffusion, network architecture plays a big role. We didn’t have any information about the interaction or connection of the users. This data would have been useful for applying information diffusion.